Information Sciences 382–383 (2017) 115–134

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Communication-reducing diffusion LMS algorithm over multitask networks

Wei Huang, Xi Yang, Guojiang Shen*

College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou, 310023, China

ARTICLE INFO

Article history:
Received 13 July 2016
Revised 17 October 2016
Accepted 6 December 2016
Available online 7 December 2016

Keywords:
Distributed estimation
Diffusion LMS
Multi-task
Mean square deviation
Communication cost

ABSTRACT

Many practical problems in the filed of distributed estimation happen to be multi-task oriented. Without prior knowledge of clustering structure, i.e. nodes do not know which clusters they belong to beforehand, distributed algorithms for parameter estimation have received great attention in recent years. In most previous work, each node collaborates with all its neighboring nodes at each iteration, which introduces unnecessary communication assumption when any node cooperates with neighboring nodes from different clusters. In this paper, we propose a novel communication-reducing diffusion LMS (Least-Mean-Square) algorithm, called the CR-dLMS algorithm, for estimating true parameters in multi-task environment. Under the CR-dLMS algorithm, we control the probabilities of data fusion from neighboring nodes by minimizing mean-square-deviation (MSD) to reduce communication cost among nodes. Theoretical analysis for the learning behavior of the CR-dLMS algorithm is performed, and simulation results show that the CR-dLMS algorithm can indeed achieve the same estimation performance as several other previous algorithms while great reducing the communication cost among nodes.

1. Introduction

In recent years, the problem of distributed estimation over sensor networks has received great concerns in theoretical studies and has a broad variety of practical applications such as environment monitoring [4], target localization [44], routing protocol design [42] and cognitive radio [20]. This is mainly because the fusion center does not necessarily exist in the network and the parameter of interest can be estimated by all nodes in cooperative way. Compared to centralized estimation, communication cost among nodes can be great reduced by using distributed strategies for parameter estimation [5,14,31,45,50].

In distributed estimation over networks, the diffusion strategy is widely adopted due to its stronger robustness to node or link failures when compared to other strategies such as the incremental strategy [31]. To date, a number of diffusion-type distributed estimation algorithms have been proposed including the diffusion Least-Mean-Square (dLMS), the diffusion Recursive-Least-Square (dRLS) [5] and the diffusion Total-Least-Square (dTLS) [26,45]. Due to the inherently simple process of LMS algorithm [13,19,48], the dLMS algorithm has been more extensively studied in a variety of scenarios, such as exploring the performances of dLMS in the presence of noisy links [22,23,53], variable step-size dLMS algorithm [21,25,39], variable component-wise step-sized LMS algorithm [18], generalized cost function and data model [9,11,15], the impact of

* Corresponding author.
E-mail address: gjshen1975@zjut.edu.cn (G. Shen).

http://dx.doi.org/10.1016/j.ins.2016.12.010
0020-0255/

network topology on the performance of dLMS algorithm [28,30], dLMS algorithm under non-Gaussian noise environment [27] and dLMS algorithm for sparse vector estimation by incorporating additional regularizers [10,29,33].

In most previous studies on the dLMS algorithm, it was assumed that all nodes share a common estimation interest. Usually, problem of this type is referred to as single-task problem. In many practical applications such as multi-target tracking [44] or spectrum estimation with multi-antenna devices [34], however, the estimation interest on different nodes may not be identical. Problem of this type is therefore referred to as multi-task problem, which can be regarded as generalization of single-task problem. In this study, we focus on multi-task problem in the context of the dLMS algorithm over adaptive networks.

Generally speaking, it is usually regarded that nodes with the same optimum estimation interest form a cluster in multi-task scenario. In [1], the authors proposed a dLMS strategy under which the nodes cooperate to estimate the true parameters that can vary in both space and time domains. In [7], the authors proposed a cooperative algorithm based diffusion adaptation to solve the multi-task learning problem, with the assumption that the estimation interest on each node consists of an offset component shared by all nodes and a node-specific component in an orthogonal subspace. In [38], under the assumption that clusters are known beforehand, the authors developed an extended dLMS algorithm for application to multi-task learning problems by minimizing mean-square error with l_2-regularization. In [47], the authors developed an optimization algorithm in which inter-cluster cooperation weights are optimized to achieve better estimation performances compared to the use of an average rule for inter-cluster cooperation. In [16], the authors proposed multi-task diffusion strategies based on the Affine Projection Algorithm (APA) to enhance the robustness against correlated input. Another group of work is described by [3,36,37], where incremental-based LMS, incremental-based RLS and diffusion-based LMS strategies are used to solve the problem of distributed estimation in scenarios where all nodes simultaneously estimate both local and global parameters.

In all the work described above, it was assumed that the information of clustering structure is known a prior, i.e. each node knows which of its neighbors share the same estimation interest as itself. Nodes in this case do not need to learn which neighbors they should cooperate with and which other nodes they should choose to ignore. In [51], the authors designed an adaptive strategy to dynamically adjust combination coefficients to endow nodes to identify clustering structure of the whole network. However, estimation performances of the algorithm in [51] are dependent on the initial conditions used by the nodes to launch their adaptation rules. In [8], the authors examined the performances of traditional dLMS algorithm with fixed combination coefficients in multi-task environment and proposed a novel unsupervised clustering method for each node collectively choosing to operate with neighboring nodes addressing the same task. Under the condition that each node does not know which other nodes share the same estimation interests, the authors of [35] studied the problem of distributed node-specific parameter estimation, i.e. scenarios in which nodes perform simultaneous estimation for both local and global unknown parameters.

Although [8] and [51] can identify clustering structure with satisfying performances via adaptively learning way, much communication cost is consumed for fusing data from neighboring nodes. In strategies proposed in [8] and [51], each node can adaptively learn to assign large combination coefficients to neighboring nodes with the same task while assigning small combination coefficients to neighboring nodes with different tasks. This requires each node to cooperate with all its neighboring nodes at each iteration. One way to lower communication cost is to reduce the opportunities of cooperation among nodes with different estimation interests, since cooperation with neighboring nodes with different estimation interest does not benefit estimation performances. In [52], the authors proposed a novel strategy that enables nodes to identify their clusters with probabilities very close to one under sufficiently small step-size, thus saving communication cost among nodes. Besides, the authors of [24] proposed a modified strategy that merges the clustering and inference tasks. Numerical simulations have shown that estimation performance can be enhanced by using the strategy in [24] than that in [52]; however, a predefined threshold for reaching a symmetrical pattern of cooperation is required for implementing these strategies.

In this paper, we propose a novel communication-reducing diffusion LMS (CR-dLMS) algorithm, to control the probabilities of data fusion from neighboring nodes by minimizing mean-square-deviation (MSD). Under the CR-dLMS algorithm, each node learns to select the neighboring nodes with which it cooperates in probabilistic and adaptive manner. The clustering structure then gradually emerges alongside the evolution of iterations. Once the adaptation process reaches the steady-state, the probability of fusing data from neighboring nodes with the same estimation interest is nontrivial - this allows nodes with the same estimation interest to cooperatively perform estimation. In the meanwhile, the probability of fusing data from neighboring nodes with different estimation interests gradually shrinks to zero. In this way, the CR-dLMS algorithm can save communication cost while still achieving satisfying estimation performances. We shall analyze the CR-dLMS algorithm in terms of its mean weight deviation and mean-square deviation in this paper. Results of numerical simulations are then followed after theoretical analysis.

The rest of paper is organized as follows. In Section 2, we introduce the proposed CR-dLMS algorithm after describing the linear data model upon which it was based. In Section 3, we present the analysis of mean-stability and learning behavior of mean-square deviations. In Section 4, we present the results of numerical simulations for comparing the CR-dLMS algorithm with several other existing algorithms. Finally, conclusions are drawn in Section 5.

Notations: In this paper, bold letters and normal letters are used to denote random quantities and deterministic quantities respectively. The superscript notation $(\cdot)^T$ stands for the transposition of a matrix or a vector, and the notation $E[\cdot]$ stands for the expectation operation. Identity matrix with size $N \times N$ is denote by I_N. The notation $\mathcal{N}_k$ denotes the neighbors of

node k, i.e. the nodes having a direct link with node k, including the node k itself. Other notations not appearing here will be defined later if necessary.

2. Algorithm description

2.1. Preliminaries

We consider a connected sensor network with N nodes spread over a given network. Node k has access to temporal stationary zero-mean measurements $\{\boldsymbol{d}_k(n), \boldsymbol{x}_{k,n}\}$, where $\boldsymbol{d}_k(n)$ is a scalar reference signal and $\boldsymbol{x}_{k,n}$ is the $1 \times L$ regression vector. In this paper, we assume both $\boldsymbol{d}_k(n)$ and $\boldsymbol{x}_{k,n}$ to be spatially and temporal independent. The covariance of $\boldsymbol{x}_{k,n}$ is given by a positive definite matrix $R_{x,k} = E\left(\boldsymbol{x}_{k,n}^{*}\boldsymbol{x}_{k,n}\right) > 0$. In the model studied, the reference signal $\boldsymbol{d}_k(n)$ and the regression vector $\boldsymbol{x}_{k,n}$ are related by the following linear equation:

$$\boldsymbol{d}_k(n) = \boldsymbol{x}_{k,n}w_{ok} + \boldsymbol{z}_k(n), \tag{1}$$

where the signal $\boldsymbol{z}_k(n)$ is the additive zero-mean Gaussian noise with variance $\sigma_{z,k}^2$ and w_{ok} is an $L \times 1$ unknown weight vector on node k which is to be estimated.

In single-task problem, all the nodes estimate the same true weight vector, i.e.

$$w_{o1} = w_{o2} = \ldots = w_{oN}. \tag{2}$$

In multi-task environment, however, the true weight vectors may vary from each other on different nodes. Currently, the linear regression model (1) is widely studied in the field of adaptive filtering because this model captures many practical problems of interests [40].

In the case of Gaussian-distribution additive noise, the dLMS algorithm was designed by minimizing the following mean-square-error cost function in distributed manner [6,32,41]:

$$J^{global}(w) = \sum_{k=1}^{N} J_k(w) = \sum_{k=1}^{N} E\left(|\boldsymbol{e}_k(n)|^2\right) = \sum_{k=1}^{N} E\left(|\boldsymbol{d}_k(n) - \boldsymbol{x}_{k,n}w|^2\right), \tag{3}$$

where $\boldsymbol{e}_k(n)$=$\boldsymbol{d}_k(n) - \boldsymbol{x}_{k,n}w$ denotes the estimation error. The process of dLMS algorithm is executed by the following two stages at each iteration:

1. **Adaptation**: Each node updates its current estimate for the true weight vector by using the LMS algorithm to obtain an intermediate estimate:
$$\boldsymbol{\psi}_k^{(n)} = \boldsymbol{w}_k^{(n-1)} + \mu_k \boldsymbol{x}_{k,n}^{*}\left[\boldsymbol{d}_k(n) - \boldsymbol{x}_{k,n}\boldsymbol{w}_k^{(n-1)}\right], \tag{4}$$
where $\mu_k > 0$ denotes the step-size for node k.
2. **Combination**: Each node aggregates its own intermediate estimate with ones from its neighboring nodes to obtain a new estimate:
$$\boldsymbol{w}_k^{(n)} = \sum_{l \in \mathcal{N}_k} c_{lk}\boldsymbol{\psi}_l^{(n)}, \tag{5}$$
where c_{lk} is the combination coefficient satisfying the following rule:
$$c_{lk} > 0 \text{ for } l \in \mathcal{N}_k \text{ and } \sum_{l \in \mathcal{N}_k} c_{lk} = 1. \tag{6}$$

In the dLMS described here, the combination stage follows the adaptation stage at each time iteration. Accordingly, this is typically referred to as the adapt-then-combine (ATC) strategy. Reversing the order of the adaptation stage and the combination stage naturally forms the combine-then-adapt (CTA) strategy. In this paper, we only utilize the ATC strategy, as it is known to consistently outperform the CTA strategy [6]. Similar analysis could be extended to the CTA strategy.

2.2. The communication-reducing dLMS (CR-dLMS) algorithm

To reduce communication cost introduced by data fusion among different nodes, we design an adaptive strategy to control the probability of fusing data from neighboring nodes. At iteration n, we define binary matrix $\boldsymbol{\Pi}^{(n)} = \left\{\boldsymbol{\pi}_{lk}^{(n)}\right\}$, $l, k = 1, 2, \ldots, N$, with $\boldsymbol{\pi}_{lk}^{(n)}$ defined as

$$\boldsymbol{\pi}_{lk}^{(n)} = \begin{cases} 1 & \text{if node } k \text{ fuses data from node } l \text{ at iteration } n, \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

Now we define the expectation of $\boldsymbol{\pi}_{lk}^{(n)}$, $E[\boldsymbol{\pi}_{lk}^{(n)}] = \boldsymbol{p}_{lk}^{(n)}$. The variable $\boldsymbol{p}_{lk}^{(n)}$ can be regarded as the probability of node k fusing an instantaneous estimate from node l at iteration n; moreover, we have $E[\boldsymbol{\pi}_{lk}^{(n)2}] = 1^2 * \boldsymbol{p}_{lk}^{(n)} + 0^2 * (1 - \boldsymbol{p}_{lk}^{(n)}) = \boldsymbol{p}_{lk}^{(n)}$.

The following modified dLMS algorithm can be obtained after accounting for probabilistic data fusion from neighboring nodes:

$$\begin{cases} \boldsymbol{\psi}_k^{(n)} = \boldsymbol{w}_k^{(n-1)} + \mu_k \boldsymbol{x}_{k,n}^* \left[\boldsymbol{d}_k(n) - \boldsymbol{x}_{k,n} \boldsymbol{w}_k^{(n-1)} \right] \\ \boldsymbol{w}_k^{(n)} = \sum\limits_{l \in \mathcal{N}_k} \xi_{lk}^{(n)} \boldsymbol{\psi}_l^{(n)} \end{cases}, \tag{8}$$

where

$$\xi_{lk}^{(n)} = \begin{cases} c_{lk} \boldsymbol{\pi}_{lk}^{(n)} & \text{if } l \neq k, \\ 1 - \sum\limits_{m \in \mathcal{N}_k \setminus \{k\}} c_{mk} \boldsymbol{\pi}_{mk}^{(n)} & \text{if } l = k. \end{cases} \tag{9}$$

The variable $\xi_{lk}^{(n)}$ for all $l,\ k = 1, 2, \ldots, N$ constitutes the matrix $\Xi^{(n)} = \left\{ \xi_{lk}^{(n)} \right\}$. The expectation of $\boldsymbol{q}_{lk}^{(n)} = E\left(\xi_{lk}^{(n)} \right)$ is:

$$\boldsymbol{q}_{lk}^{(n)} = \begin{cases} c_{lk} \boldsymbol{p}_{lk}^{(n)} & \text{if } l \neq k, \\ 1 - \sum\limits_{m \in \mathcal{N}_k \setminus \{k\}} c_{mk} \boldsymbol{p}_{mk}^{(n)} & \text{if } l = k. \end{cases} \tag{10}$$

All $\boldsymbol{q}_{lk}^{(n)}$ for $l,\ k = 1, 2, \ldots, N$ constitute the matrix $\boldsymbol{Q}^{(n)} = \left\{ \boldsymbol{q}_{lk}^{(n)} \right\}$. Clearly, $E\left[\Xi^{(n)} \right] = \boldsymbol{Q}^{(n)}$.

To find the optimal probability of data fusion from neighboring nodes at each iteration, we seek to optimize the following instantaneous MSD on node k and at iteration n:

$$\mathrm{MSD}_k(n) = E\left(\left\| w_{ok} - \sum_{l \in \mathcal{N}_k} \xi_{lk}^{(n)} \boldsymbol{\psi}_l^{(n)} \right\|^2 \right). \tag{11}$$

By defining $\xi_k^{(n)} = \left[\xi_{1k}^{(n)}, \ldots, \xi_{Nk}^{(n)} \right]^T$, Eq. (11) can be formulated as follows:

$$\begin{aligned} &\min_{\xi_k^{(n)}} \sum_{l=1}^{N} E\left[\xi_k^{(n)*} \boldsymbol{\Psi}_k(n) \xi_k^{(n)} \right] \\ &\text{subject to } \mathbf{1}^T \xi_k(n) = 1,\ \xi_{lk}(n) \geq 0,\ \xi_{lk}(n) = 0 \text{ if } l \notin \mathcal{N}_k. \end{aligned} \tag{12}$$

where $\boldsymbol{\Psi}_k(n)$ is the matrix on node k with the (l, p)th entry defined as follows:

$$[\boldsymbol{\Psi}_k(n)]_{lp} = \begin{cases} \left[w_{ok} - \boldsymbol{\psi}_l(n) \right]^* \left[w_{ok} - \boldsymbol{\psi}_p(n) \right] & l, p \in \mathcal{N}_k \\ 0 & \text{otherwise} \end{cases}. \tag{13}$$

In order to make the optimization problem (12) tractable, we drop the off-diagonal entries in matrix $\boldsymbol{\Psi}_k(n)$ [8]. The problem (12) can then be simplified as follows:

$$\begin{aligned} &\min_{\xi_k^{(n)}} \sum_{l=1}^{N} E\left[\xi_{lk}^{(n)2} \left\| w_{ok} - \boldsymbol{\psi}_l^{(n)} \right\|^2 \right] \\ &\text{subject to } \mathbf{1}^T \xi_k^{(n)} = 1,\ \xi_{lk}^{(n)} \geq 0,\ \xi_{lk}^{(n)} = 0 \text{ if } l \notin \mathcal{N}_k. \end{aligned} \tag{14}$$

Similar to [43], we assume $\xi_k^{(n)}$ and $\boldsymbol{\psi}_l^{(n)}$ are independent of each other. Accordingly,

$$\sum_{l=1}^{N} E\left[\xi_{lk}^{(n)2} \left\| w_{ok} - \boldsymbol{\psi}_l^{(n)} \right\|^2 \right] = \sum_{l=1}^{N} E\left[\xi_{lk}^{(n)2} \right] E\left[\left\| w_{ok} - \boldsymbol{\psi}_l^{(n)} \right\|^2 \right]. \tag{15}$$

Using instantaneous value to approximate the expectation $E\left[||w_{ok} - \boldsymbol{\psi}_l^{(n)}||^2 \right]$, then

$$\begin{aligned} &\sum_{l=1}^{N} E\left[\xi_{lk}^{(n)2} \right] E\left[\left\| w_{ok} - \boldsymbol{\psi}_l^{(n)} \right\|^2 \right] \\ &\approx \sum_{l=1}^{N} E\left[\xi_{lk}^{(n)2} \right] \left\| w_{ok} - \boldsymbol{\psi}_l^{(n)} \right\|^2 \\ &= \sum_{l \in \mathcal{N}_k / k} \boldsymbol{p}_{lk}^{(n)} c_{lk}^2 \left\| w_{ok} - \boldsymbol{\psi}_l^{(n)} \right\|^2 \end{aligned}$$

$$+\left\|w_{ok}-\boldsymbol{\psi}_k^{(n)}\right\|^2*\left(1-\sum_{l\in\mathcal{N}_k\backslash k}\boldsymbol{p}_{lk}^{(n)}c_{lk}\right)^2$$
$$+\left\|w_{ok}-\boldsymbol{\psi}_k^{(n)}\right\|^2*\sum_{l\in\mathcal{N}_k\backslash k}\left(\boldsymbol{p}_{lk}^{(n)}-\boldsymbol{p}_{lk}^{(n)2}\right)c_{lk}^2. \tag{16}$$

The last equation comes from $E\left[\boldsymbol{\pi}_{lk}^{(n)}\right]=\boldsymbol{p}_{lk}^{(n)}$, and

$$E\left[\boldsymbol{\pi}_{lk}^{(n)}\boldsymbol{\pi}_{mk}^{(n)}\right]=\begin{cases}\boldsymbol{p}_{lk}^{(n)}\boldsymbol{p}_{mk}^{(n)} & l\neq m,\\ \boldsymbol{p}_{lk}^{(n)} & l=m.\end{cases} \tag{17}$$

Eq. (16) necessitates a proper approximation $\hat{\boldsymbol{w}}_{ok}^{(n)}$ for w_{ok} at iteration n since w_{ok} is the unknown weight vector being sought. A reasonable approximation for w_{ok} is:

$$\hat{\boldsymbol{w}}_{ok}^{(n)}=\boldsymbol{\psi}_k^{(n)}-\mu_k\nabla_k J(w)|_{\boldsymbol{\psi}_k(n)}\approx\boldsymbol{\psi}_k^{(n)}+\mu_k\boldsymbol{x}_{k,n}^*\left[\boldsymbol{d}_k(n)-\boldsymbol{x}_{k,n}\boldsymbol{\psi}_k^{(n)}\right]. \tag{18}$$

This approximation has successfully been utilized in [8] for calculating the dynamical combination matrix.

Define $\boldsymbol{p}_k^{(n)}$ as a $1\times(|\mathcal{N}_k|-1)$ vector consisting of all $\boldsymbol{p}_{lk}^{(n)}$, $l\in\mathcal{N}_k\backslash k$. Problem (14) then becomes the following constrained optimization problem, which is convex and can be solved by some adaptive algorithm to find the optimal solution:

$$\begin{aligned}\min_{\boldsymbol{p}_k^{(n)}} f\left(\boldsymbol{p}_k^{(n)}\right)=&\min_{\boldsymbol{p}_k^{(n)}}\sum_{l\in\mathcal{N}_k\backslash k}\boldsymbol{p}_{lk}^{(n)}c_{lk}^2\left\|\hat{\boldsymbol{w}}_{ok}^{(n)}-\boldsymbol{\psi}_l^{(n)}\right\|^2\\ &+\left[\left(1-\sum_{l\in\mathcal{N}_k\backslash k}\boldsymbol{p}_{lk}^{(n)}c_{lk}\right)^2+\sum_{l\in\mathcal{N}_k\backslash k}\left(\boldsymbol{p}_{lk}^{(n)}-\boldsymbol{p}_{lk}^{(n)2}\right)c_{lk}^2\right]*\left\|\hat{\boldsymbol{w}}_{ok}^{(n)}-\boldsymbol{\psi}_k^{(n)}\right\|^2\\ &s.t.\ 0\le\boldsymbol{p}_{lk}\le 1,\ l\in\mathcal{N}_k\backslash k.\end{aligned} \tag{19}$$

Since the probability of data fusion from neighboring nodes lies in the range [0, 1], we use the hybrid steepest-descent method recursively to determine the optimal control for data fusion probability. Now we define $\nabla_{\boldsymbol{p}_k}f(\boldsymbol{p}_k^{(n)})$ as the gradient for $f(\boldsymbol{p}_k^{(n)})$ and $[\nabla_{\boldsymbol{p}_k}f(\boldsymbol{p}_k^{(n)})]_l$ as the component corresponding to node l. The expression $[\nabla_{\boldsymbol{p}_k}f(\boldsymbol{p}_k^{(n)})]_l$ can be defined as follows:

$$\left[\nabla_{\boldsymbol{p}_k}f\left(\boldsymbol{p}_k^{(n)}\right)\right]_l=c_{lk}^2\left\|\hat{\boldsymbol{w}}_{ok}^{(n)}-\boldsymbol{\psi}_l^{(n)}\right\|^2+\left[-2c_{lk}\left(1-\sum_{l\in\mathcal{N}_k\backslash k}\boldsymbol{p}_{lk}^{(n)}c_{lk}\right)+\left(1-2\boldsymbol{p}_{lk}^{(n)}\right)c_{lk}^2\right]*\left\|\hat{\boldsymbol{w}}_{ok}^{(n)}-\boldsymbol{\psi}_k^{(n)}\right\|^2. \tag{20}$$

Then, we implement the following hybrid steepest-descent method for the optimization problem:

$$\boldsymbol{p}_k^{(n)}=res\left[\boldsymbol{p}_k^{(n-1)}-\varepsilon_k\nabla_{\boldsymbol{p}_k}f\left(\boldsymbol{p}_k^{(n-1)}\right)\right]. \tag{21}$$

In Eq. (21), ε_k is the step-size for adapting $\boldsymbol{p}_k^{(n)}$. The operation $res[*]$ restricts each component of a vector in the range [0, 1], i.e.

$$res[x]_i=\max\left(0,\min\left(x_i,1\right)\right), \tag{22}$$

where x is a vector, x_i is the ith component of x, and $res[x]_i$ denotes the ith component of $res[x]$. In Eq. (20), $\boldsymbol{\psi}_l(n)$ denotes the intermediate estimate on the neighboring node l at iteration n. However, node l does not necessarily share $\boldsymbol{\psi}_l^{(n)}$ to node k at each iteration because data fusion is performed only probabilistically. To solve this problem, the node k must store the intermediate estimate sent from each neighboring node l at the latest iteration. If a new intermediate estimate $\boldsymbol{\psi}_l^{(n)}$ arrives at node k at iteration n, the corresponding intermediate estimate stored on node k is replaced by $\boldsymbol{\psi}_l^{(n)}$. Otherwise, the corresponding intermediate estimate stored on node k remains unchanged. For convenience, the intermediate estimate from node l stored on node k at iteration n is denoted by $\widehat{\psi}_{lk}^{(n)}$. Therefore, we can use $\widehat{\psi}_{lk}^{(n)}$ to represent $\boldsymbol{\psi}_l^{(n)}$ in Eq. (20).

To illustrate our algorithm more clearly, a flowchart for the CR-dLMS algorithm is provided in Fig. 1 and a summary for the CR-dLMS algorithm is provided in Algorithm 1.

Remark: We note in [17], the authors proposed the partial diffusion affine projection algorithm (PD-APA), in which clustering structure is known a prior, and the communication cost is reduced by virtue of the fact that each node transmits only a subset of all entries of its intermediate estimate to its neighborhood. In our proposed CR-dLMS algorithm, however, each node can gradually choose to cooperate only with its neighbors in the same cluster rather than all its neighbors in adaptation process without requiring priori information of clustering structure. In short, CR-dLMS reduces communication cost from a different approach than PD-APA.

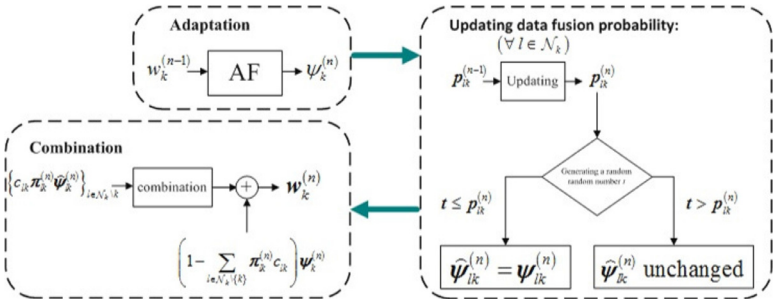

Fig. 1. Flowchart for the proposed CR-dLMS algorithm.

Algorithm 1 CR-dLMS algorithm.

Initializing:

For each node k, we set $\boldsymbol{w}_k^{(-1)} = 0$.The initial probability $\boldsymbol{p}_{lk}^{(-1)}$ of data fusion from any neighboring node $l \in \mathcal{N}_k\backslash k$ is set to one. In addition, we set $\widehat{\psi}_{lk}^{(-1)} = 0$ for any $l \in \mathcal{N}_k\backslash k$. Then, for each node k, the following three steps are repeated at each iteration $n \geq 0$:

1. Adaptation step:

$$\boldsymbol{\psi}_k^{(n)} = \boldsymbol{w}_k^{(n-1)} + \mu_k \boldsymbol{x}_{k,n}^* \left[\boldsymbol{d}_k(n) - \boldsymbol{x}_{k,n}\boldsymbol{w}_k^{(n-1)}\right]. \quad (23)$$

2. **Updating data fusion probability:**

$$\begin{cases} \left[\nabla_{\boldsymbol{p}_k} f\left(\boldsymbol{p}_k^{(n-1)}\right)\right]_l \\ = c_{lk}^2 \left\| \hat{\boldsymbol{w}}_{ok}^{(n-1)} - \widehat{\psi}_{lk}^{(n-1)} \right\|^2 + \left[-2c_{lk}\left(1 - \sum_{l\in\mathcal{N}_k\backslash k} \boldsymbol{p}_{lk}^{(n-1)} c_{lk}\right)\right. \\ \left. + \left(1 - 2\boldsymbol{p}_{lk}^{(n-1)}\right)c_{lk}^2\right] * \left|\left|\hat{\boldsymbol{w}}_{ok}^{(n-1)} - \boldsymbol{\psi}_k^{(n-1)}\right|\right|^2 \\ \boldsymbol{p}_k^{(n)} = res\left[\boldsymbol{p}_k^{(n-1)} - \varepsilon_k \nabla_{\boldsymbol{p}_k} f\left(\boldsymbol{p}_k^{(n-1)}\right)\right]. \end{cases} \quad (24)$$

where $\hat{\boldsymbol{w}}_{ok}^{(n)}$ is the approximation at iteration $n$for the true weight vectorw_{ok}, which is given by:

$$\hat{\boldsymbol{w}}_{ok}^{(n)} = \boldsymbol{\psi}_k^{(n)} - \mu_k \nabla_k J(w)|_{\boldsymbol{\psi}_k^{(n)}} \approx \boldsymbol{\psi}_k^{(n)} + \mu_k \boldsymbol{x}_{k,n}^* \left[\boldsymbol{d}_k(n) - \boldsymbol{x}_{k,n}\boldsymbol{\psi}_k^{(n)}\right]. \quad (25)$$

3. Update $\widehat{\psi}_{lk}^{(n)}$ if node k chooses to fuse $\boldsymbol{\psi}_l^{(n)}$ from node l. Otherwise, $\widehat{\psi}_{lk}^{(n)}$ keeps unchanged.
4. **Combination step**: According to the updated $\boldsymbol{p}_k^{(n)}$, determine the value of $\boldsymbol{\pi}_{lk}^{(n)}$, $l \in \mathcal{N}_k\backslash k$. Then, the combination step is performed as following:

$$\boldsymbol{w}_k^{(n)} = \left(1 - \sum_{l\in\mathcal{N}_k\backslash\{k\}} \boldsymbol{\pi}_{lk}^{(n)} c_{lk}\right)\boldsymbol{\psi}_k^{(n)} + \sum_{l\in\mathcal{N}_k\backslash\{k\}} \boldsymbol{\pi}_{lk}^{(n)} c_{lk} \widehat{\psi}_{lk}^{(n)}. \quad (26)$$

3. Performance analysis

In this section, the performances of CR-dLMS algorithm in multi-task environment are analyzed in terms of mean stability and learning behavior of mean-square error. For performance analysis, we define the weight error vector for the node k:

$$\tilde{\boldsymbol{w}}_k^{(n)} = \boldsymbol{w}_k^{(n)} - w_{ok}. \quad (27)$$

To investigate the global performance over the whole network, we collect the data on all nodes into block vectors and matrices to form the following global variables:

$$\begin{aligned} & w_o = col\{w_{o1}, \ldots, w_{oN}\},\ \boldsymbol{x}_n = diag\{\boldsymbol{x}_{1,n}, \ldots, \boldsymbol{x}_{N,n}\}, \\ & \boldsymbol{w}^{(n)} = col\left\{\boldsymbol{w}_1^{(n)}, \ldots, \boldsymbol{w}_N^{(n)}\right\},\ \boldsymbol{\psi}^{(n)} = col\left\{\boldsymbol{\psi}_1^{(n)}, \ldots, \boldsymbol{\psi}_N^{(n)}\right\}, \\ & \tilde{\boldsymbol{w}}^{(n)} = col\left\{\tilde{\boldsymbol{w}}_1^{(n)}, \ldots, \tilde{\boldsymbol{w}}_N^{(n)}\right\},\ U = diag\{\mu_1 I_L, \ldots, \mu_N I_L\}, \\ & \boldsymbol{p}_{zx,n} = col\left\{\boldsymbol{x}_{1,n}^* \boldsymbol{z}_1(n), \ldots, \boldsymbol{x}_{N,n}^* \boldsymbol{z}_N(n)\right\}, \\ & \boldsymbol{K}^{(n)} = \boldsymbol{\Xi}^{(n)} \otimes I_L,\ \boldsymbol{S}^{(n)} = \boldsymbol{Q}^{(n)} \otimes I_L, \\ & R_x = diag\{R_{x,1}, \ldots, R_{x,N}\}. \end{aligned} \quad (28)$$

In Eq. (28), the operation $col\{\cdot\}$ stacks all column vectors in the curly braces and $\otimes$ denotes the Kronecker product.

To facilitate performance analysis, we make the following assumptions:

Assumption 1. Both the additive noises $\boldsymbol{z}_k(n)$ and regressors $\boldsymbol{x}_{k,n}$, $\forall k, n$, are spatially and temporally independent. In addition, $\boldsymbol{z}_k(n)$ and $\boldsymbol{x}_{k,n}$, $\forall k, n$, are independent of each other.

Assumption 2. The regression vector $\boldsymbol{x}_{k,n}$ is independent of $\tilde{\boldsymbol{w}}_l^{(m)}$ for all l and $m < n$.

Assumption 3. At any iteration $n \geq 0$, the matrix $K^{(n)}$ is independent of both $\boldsymbol{x}_n$ and $\tilde{\boldsymbol{w}}^{(n-1)}$.

The assumptions above have been widely applied in the adaptive learning literature when performing theoretical analysis, such as [6,32,43]. Assumption 3 in particular has been used in a similar context [43], where the authors assumed the dynamical weight matrix C_i to be independent of both the regression vector and the weight error vector.

3.1. Mean stability analysis

The mean stability analysis is performed here to find sufficient condition under which the estimate can converge to the true weight vector. From Eq. (23), the estimation error can be rewritten as follows:

$$\boldsymbol{d}_k(n) - \boldsymbol{x}_{k,n}\boldsymbol{w}_k^{(n-1)} = \boldsymbol{z}_k(n) - \boldsymbol{x}_{k,n}\left[\boldsymbol{w}_k^{(n-1)} - w_{ok}\right]. \tag{29}$$

By subtracting the true weight vector from both sides of Eq. (23) and using Eq. (29), we have

$$\begin{aligned}\boldsymbol{\psi}_k^{(n)} - w_{ok} &= \left[\boldsymbol{w}_k^{(n-1)} - w_{ok}\right] + \mu_k \boldsymbol{x}_{k,n}^* \boldsymbol{z}_k(n) - \mu_k \boldsymbol{x}_{k,n}^* \boldsymbol{x}_{k,n}\left[\boldsymbol{w}_k^{(n-1)} - w_{ok}\right] \\ &= \tilde{\boldsymbol{w}}_k^{(n-1)} + \mu_k \boldsymbol{x}_{k,n}^* \boldsymbol{z}_k(n) - \mu_k \boldsymbol{x}_{k,n}^* \boldsymbol{x}_{k,n} \tilde{\boldsymbol{w}}_k^{(n-1)}.\end{aligned} \tag{30}$$

The block-based version of Eq. (30) for evaluating the weight errors on all nodes over the whole network is:

$$\boldsymbol{\psi}^{(n)} - w_o = \tilde{\boldsymbol{w}}^{(n-1)} + U\boldsymbol{p}_{zx,n} - U\boldsymbol{x}_n^*\boldsymbol{x}_n\tilde{\boldsymbol{w}}^{(n-1)} = [I_{NL} - U\boldsymbol{x}_n^*\boldsymbol{x}_n]\tilde{\boldsymbol{w}}^{(n-1)} + U\boldsymbol{p}_{zx,n}. \tag{31}$$

To facilitate mean and mean-square behavior analysis, we replace $\widehat{\psi}_{lk}^{(n)}$ with $\boldsymbol{\psi}_l^{(n)}$ in Eq. (26) This approximation is reasonable especially at the later stage of adaption process. If nodes l and k have different tasks, the small data fusion probability from node l to k may result in large deviation between $\widehat{\psi}_{lk}^{(n)}$ and $\boldsymbol{\psi}_l^{(n)}$ at the later stage of adaptation process. Ensuring small data fusion probability from node l to node k can mitigate the effect of any such large deviation between $\widehat{\psi}_{lk}^{(n)}$ and $\boldsymbol{\psi}_l^{(n)}$. Therefore, the block-based version of Eq. (26) can be approximately expressed as:

$$\boldsymbol{w}^{(n)} \approx K^{(n)*}\boldsymbol{\psi}^{(n)}. \tag{32}$$

Subtracting w_o from both sides of Eq. (32) and using Eq. (31) yields:

$$\begin{aligned}\tilde{\boldsymbol{w}}^{(n)} &\approx K^{(n)*}\boldsymbol{\psi}^{(n)} - w_o \\ &= K^{(n)*}\left[\boldsymbol{\psi}^{(n)} - w_o\right] + \left[K^{(n)*} - I_{NL}\right]w_o \\ &= K^{(n)*}[I_{NL} - U\boldsymbol{x}_n^*\boldsymbol{x}_n]\tilde{\boldsymbol{w}}^{(n-1)} + K^{(n)*}U\boldsymbol{p}_{zx,n} + \left[K^{(n)*} - I_{NL}\right]w_o.\end{aligned} \tag{33}$$

We define

$$\boldsymbol{B}^{(n)} = \boldsymbol{K}^{(n)*}[I_{NL} - U\boldsymbol{x}_n^*\boldsymbol{x}_n], \boldsymbol{g}^{(n)} = \boldsymbol{K}^{(n)*}U\boldsymbol{p}_{zx,n}, \boldsymbol{r}^{(n)} = \left[\boldsymbol{K}^{(n)*} - I_{NL}\right]w_o, \tag{34}$$

so that Eq. (33) can be rewritten as

$$\tilde{\boldsymbol{w}}^{(n)} = \boldsymbol{B}^{(n)}\tilde{\boldsymbol{w}}^{(n-1)} + \boldsymbol{g}^{(n)} + \boldsymbol{r}^{(n)}. \tag{35}$$

In Eq. (35), the time-varying term $\boldsymbol{r}^{(n)}$ vanishes in single-task scenario. However, in multi-task scenario, the term $\boldsymbol{r}^{(n)}$ is non-zero and makes performance analysis more complex than that under the single-task case.

Following Assumption 3, the expectation of $\boldsymbol{B}^{(n)}$ can be expressed as

$$E\left[\boldsymbol{B}^{(n)}\right] = \boldsymbol{B}_e^{(n)} = \boldsymbol{S}^{(n)*}(I_{NL} - UR_x), \tag{36}$$

where $\boldsymbol{S}^{(n)} = E\left[\boldsymbol{K}^{(n)}\right]$. Since the zero-mean additive noise is independent of any other signal, we have $E\left[\boldsymbol{g}^{(n)}\right] = 0$. In addition, the expectation of $\boldsymbol{r}^{(n)}$ can be expressed as:

$$E\left[\boldsymbol{r}^{(n)}\right] = \boldsymbol{r} = \left[\boldsymbol{S}^{(n)*} - I_{NL}\right]w_o. \tag{37}$$

Based on Assumption 2 and Assumption 3, the expectation of the weight vector error is:

$$E\left[\tilde{\boldsymbol{w}}^{(n)}\right] = \boldsymbol{S}^{(n)*}[I_{NL} - UR_x]E\left[\tilde{\boldsymbol{w}}^{(n-1)}\right] + \left[\boldsymbol{S}^{(n)*} - I_{NL}\right]w_o. \tag{38}$$

Now we present the following theorem for the convergence of the CR-dLMS algorithm in the mean sense:

Theorem 1. *(Stability in the mean) For the data model (1), and based on Assumptions 1, 2 and 3, the CR-dLMS algorithm applied to the multi-task networks asymptotically converges in the mean if the step-sizes satisfy the following condition:*

$$0 < \mu_k < \frac{2}{\lambda_{\max}\left(R_{x,k}\right)} \text{ for all } k = 1, \ldots N, \tag{39}$$

where $\lambda_{\max}(\cdot)$ denotes the maximum eigenvalue of the matrix argument.

Proof. First, since $\boldsymbol{S}^{(n)}$ is left-stochastic, the spectrum norm of $\boldsymbol{S}^{(n)*}[I_{NL} - UR_x]$ is upper bounded by that of $I_{NL} - UR_x$. Thus, it is required that

$$|\lambda_{\max}(I_{NL} - UR_x)| < 1. \tag{40}$$

On the other hand, the set of eigenvalues of $(I_{NL} - UR_x)$ is the union of the sets of eigenvalues of $\left(I_L - \mu_k R_{x,k}\right)$ for all $k \in \{1, \ldots, N\}$; so Eq. (40) is satisfied if

$$\left|\lambda_{\max}\left(I_L - UR_{x,k}\right)\right| < 1. \tag{41}$$

Considering $\left[\boldsymbol{S}^{(n)*} - I_{NL}\right]w_o$ is clearly bounded, according to Eq. (41), we can conclude that the step-size on each node k must at least satisfy $0 < \mu_k < 2/\lambda_{\max}(R_{x,\,k})$. □

3.2. Mean-square error behavior analysis

In order to evaluate mean-square performances of the CR-dLMS algorithm, we define the mean square deviation (MSD),

$$\eta_k^{(n)} = E\left\|\tilde{\boldsymbol{w}}_k^{(n-1)}\right\|^2, \tag{42}$$

and the excess mean square error (EMSE),

$$\zeta_k^{(n)} = E\left\|\boldsymbol{x}_{k,n}\tilde{\boldsymbol{w}}_k^{(n-1)}\right\|^2. \tag{43}$$

Averaging the two quantities over all nodes in the network, the network MSD and EMSE can be obtained:

$$\eta^{(n)} = \frac{1}{N}\sum_{k=1}^{N}\eta_k^{(n)} \text{ and } \zeta^{(n)} = \frac{1}{N}\sum_{k=1}^{N}\zeta_k^{(n)}. \tag{44}$$

To characterize mean-square behavior of the CR-dLMS algorithm under a unified framework, we define the mean-square of the weight error vector weighted by any positive-definite matrix Σ, i.e.

$$E\|\tilde{\boldsymbol{w}}(n)\|_{\Sigma}^2 = \tilde{\boldsymbol{w}}^*(n)\Sigma\tilde{\boldsymbol{w}}(n). \tag{45}$$

The matrix Σ can be freely chosen to arrive at various mean-square behaviors. For example, choosing $\Sigma = I_{NL}/N$ leads to the network MSD while the network EMSE can be obtained by assigning

$$\Sigma = R_x/N. \tag{46}$$

To facilitate performance analysis for the mean-square error behavior, we introduce the following variable transformations:

$$\begin{aligned} &\bar{\boldsymbol{w}}^{(n)} = T^*\tilde{\boldsymbol{w}}^{(n)}, \qquad \bar{\boldsymbol{x}}^{(n)} = \boldsymbol{x}^{(n)}T, \qquad \bar{w}_o = T^*w_o, \qquad \bar{\boldsymbol{K}}^{(n)} = T^*\boldsymbol{K}^{(n)}T \\ &\bar{\Sigma} = T^*\Sigma T, \qquad \boldsymbol{p}_{z\bar{x},n} = T^*\boldsymbol{p}_{zx,n}, \qquad \bar{\boldsymbol{S}}^{(n)} = T^*\boldsymbol{S}^{(n)}T. \end{aligned} \tag{47}$$

In Eq. (47) above, the matrix Σ is an arbitrary $NL \times NL$ matrix that can be freely chosen. The matrix T comes from the decomposition $R_x = T\Lambda T^*$, where $\Lambda = diag\{\Lambda_1, \ldots, \Lambda_N\}$ is a diagonal matrix. Under the transformation (47), the matrix $E[\bar{\boldsymbol{x}}_n^*\bar{\boldsymbol{x}}_n]$ is a diagonal matrix. Moreover, for the diagonal matrix U, we have $T^*U = UT^*$.

Pre-multiplying T^* on both sides of Eq. (33) yields:

$$\bar{\boldsymbol{w}}^{(n)} = \bar{\boldsymbol{K}}^{(n)*}[I_{NL} - U\bar{\boldsymbol{x}}_n^*\bar{\boldsymbol{x}}_n]\bar{\boldsymbol{w}}^{(n-1)} + \bar{\boldsymbol{K}}^{(n)*}U\boldsymbol{p}_{z\bar{x},n} + [\bar{\boldsymbol{K}}^{(n)*} - I_{NL}]\bar{w}_o. \tag{48}$$

Then, under Assumption 2 and Assumption 3, the mean-square of the weight error vector $\bar{\boldsymbol{w}}^{(n)}$ weighted by the positive-definite matrix $\bar{\Sigma}$, is given by:

$$\begin{aligned} E\left\|\bar{\boldsymbol{w}}^{(n)}\right\|_{\bar{\Sigma}}^2 = {} & E\left(\left\|\bar{\boldsymbol{w}}^{(n-1)}\right\|_{\bar{\boldsymbol{\Sigma}}^{(n)}}^2\right) + E\left[\boldsymbol{p}_{z\bar{x},n}^*U\bar{\boldsymbol{K}}^{(n)}\bar{\Sigma}\bar{\boldsymbol{K}}^{(n)*}U\boldsymbol{p}_{z\bar{x},n}\right] \\ & + E\left[\bar{w}_o^*\left(\bar{\boldsymbol{K}}^{(n)} - I_{NL}\right)\bar{\Sigma}\left(\bar{\boldsymbol{K}}^{(n)*} - I_{NL}\right)\bar{w}_o\right] \\ & + E\left[\bar{\boldsymbol{w}}^{(n-1)*}\left(I_{NL} - U\bar{\boldsymbol{x}}_n^*\bar{\boldsymbol{x}}_n\right)\bar{\boldsymbol{K}}^{(n)}\bar{\Sigma}\left[\bar{\boldsymbol{K}}^{(n)*} - I_{NL}\right]\bar{w}_o\right] \\ & + E\left[\bar{w}_o^*\left(\bar{\boldsymbol{K}}^{(n)} - I_{NL}\right)\bar{\Sigma}\bar{\boldsymbol{K}}^{(n)*}\left(I_{NL} - U\bar{\boldsymbol{x}}_n^*\bar{\boldsymbol{x}}_n\right)\bar{\boldsymbol{w}}^{(n-1)}\right], \end{aligned} \tag{49}$$

where $\bar{\boldsymbol{\Sigma}}^{(n)}$ is the $NL \times NL$ random matrix defined as

$$\bar{\boldsymbol{\Sigma}}^{(n)} = E\left[\left(I_{NL} - U\bar{\boldsymbol{x}}_n^*\bar{\boldsymbol{x}}_n\right)^*\bar{\boldsymbol{K}}^{(n)}\bar{\Sigma}\bar{\boldsymbol{K}}^{(n)*}\left(I_{NL} - U\bar{\boldsymbol{x}}_n^*\bar{\boldsymbol{x}}_n\right)\right] = E\left[\bar{\boldsymbol{B}}^{(n)*}\bar{\Sigma}\bar{\boldsymbol{B}}^{(n)}\right]. \tag{50}$$

Here,

$$\bar{\boldsymbol{B}}^{(n)} = \bar{\boldsymbol{K}}^{(n)*}[I_{NL} - U\bar{\boldsymbol{x}}_n^*\bar{\boldsymbol{x}}_n]. \tag{51}$$

Following Assumption 3, we have

$$E\left[\bar{\boldsymbol{B}}^{(n)}\right] = \bar{\boldsymbol{S}}^{(n)*}[I_{NL} - U\Lambda]. \tag{52}$$

Defining the operator *vec*{} that stacks all columns of a matrix on top of each other, the matrices $\bar{\Sigma}$ and $\bar{\boldsymbol{\Sigma}}^{(n)}$ are vectorized as follows:

$$\bar{\sigma} = vec\left\{\bar{\Sigma}\right\} \text{ and } \bar{\boldsymbol{\sigma}}^{(n)} = vec\left\{\bar{\boldsymbol{\Sigma}}^{(n)}\right\}. \tag{53}$$

Considering the relationship (50) between $\bar{\Sigma}$ and $\bar{\boldsymbol{\Sigma}}^{(n)}$, $\bar{\boldsymbol{\sigma}}^{(n)}$ and $\bar{\sigma}$ can be related by

$$\bar{\boldsymbol{\sigma}}^{(n)} = \bar{\boldsymbol{H}}^{(n)}\bar{\sigma}, \tag{54}$$

where $\bar{\boldsymbol{H}}^{(n)}$ is an $(NL)^2 \times (NL)^2$ matrix defined as

$$\bar{\boldsymbol{H}}^{(n)} = E\left[\bar{\boldsymbol{B}}^{(n)*} \otimes \bar{\boldsymbol{B}}^{(n)*}\right]. \tag{55}$$

For sufficiently small step-sizes, $\bar{\boldsymbol{H}}^{(n)}$ can be approximated by

$$\bar{\boldsymbol{H}}^{(n)} \approx \left[(I_{NL} - U\Lambda)\bar{\boldsymbol{S}}^{(n)}\right] \otimes \left[(I_{NL} - U\Lambda)\bar{\boldsymbol{S}}^{(n)}\right], \tag{56}$$

i.e. high powers of sufficiently small step-sizes can be ignored. This approximation has been previously adopted in [8,41] and does not have any substantial impact on the conclusion.

In the RHS of Eq. (49), except for the term $E\left(\left\|\bar{\boldsymbol{w}}^{(n-1)}\right\|^2_{\bar{\boldsymbol{\Sigma}}^{(n)}}\right)$, all other four terms are evaluated in Appendix A.

Then, with Eq. (A.2), (A.3), (A.4), (A.5) in Appendix A, Eq. (49) allows us to finally obtain the following recursion for the mean-square of weight error vector:

$$\begin{aligned} E\left\|\bar{\boldsymbol{w}}^{(n)}\right\|^2_{\bar{\sigma}} &\approx E\left(\left\|\bar{\boldsymbol{w}}^{(n-1)}\right\|^2_{\bar{\boldsymbol{H}}^{(n)}\bar{\sigma}}\right) + \left[\bar{\boldsymbol{b}}_1^{(n)*} + \bar{\boldsymbol{b}}_2^{(n)*} + \bar{\boldsymbol{b}}_3^{(n)*} + \bar{\boldsymbol{b}}_4^{(n)*}\right]\bar{\sigma} \\ &= E\left(\left\|\bar{\boldsymbol{w}}^{(n-1)}\right\|^2_{\bar{\boldsymbol{H}}^{(n)}\bar{\sigma}}\right) + \bar{\boldsymbol{b}}^{(n)*}\bar{\sigma}, \end{aligned} \tag{57}$$

where

$$\bar{\boldsymbol{b}}^{(n)} = \bar{\boldsymbol{b}}_1^{(n)} + \bar{\boldsymbol{b}}_2^{(n)} + \bar{\boldsymbol{b}}_3^{(n)} + \bar{\boldsymbol{b}}_4^{(n)}. \tag{58}$$

Next, we shall derive the learning behavior of weighted mean-square of weight error vector. By initialing $\boldsymbol{w}_k^{(-1)} = 0$ for each node k and iterating Eq. (57), we have

$$\begin{aligned} &E\left\|\bar{\boldsymbol{w}}^{(n)}\right\|^2_{\bar{\sigma}} \approx E\left(\left\|\bar{\boldsymbol{w}}^{(n-1)}\right\|^2_{\bar{\boldsymbol{H}}^{(n)}\bar{\sigma}}\right) + \bar{\boldsymbol{b}}^{(n)*}\bar{\sigma}, \\ &E\left(\left\|\bar{\boldsymbol{w}}^{(n-1)}\right\|^2_{\bar{H}(n)\bar{\sigma}}\right) \approx E\left(\left\|\bar{\boldsymbol{w}}^{(n-2)}\right\|^2_{\bar{\boldsymbol{H}}^{(n-1)}\bar{\boldsymbol{H}}^{(n)}\bar{\sigma}}\right) + \bar{\boldsymbol{b}}^{(n-1)*}\bar{\boldsymbol{H}}^{(n)}\bar{\sigma}, \\ &\cdots \\ &E\left(\left\|\bar{\boldsymbol{w}}^{(1)}\right\|^2_{\bar{\boldsymbol{H}}^{(2)}\bar{\boldsymbol{H}}^{(3)}\cdots\bar{\boldsymbol{H}}^{(n)}\bar{\sigma}}\right) \approx E\left(\left\|\bar{\boldsymbol{w}}^{(0)}\right\|^2_{\bar{\boldsymbol{H}}^{(1)}\bar{\boldsymbol{H}}^{(2)}\cdots\bar{\boldsymbol{H}}^{(n)}\bar{\sigma}}\right) + \bar{\boldsymbol{b}}^{(1)*}\bar{\boldsymbol{H}}^{(2)}\bar{\boldsymbol{H}}^{(3)}\cdots\bar{\boldsymbol{H}}^{(n)}\bar{\sigma}, \\ &E\left(\left\|\bar{\boldsymbol{w}}^{(0)}\right\|^2_{\bar{\boldsymbol{H}}^{(1)}\bar{\boldsymbol{H}}^{(2)}\cdots\bar{\boldsymbol{H}}^{(n)}\bar{\sigma}}\right) \approx \|\bar{w}_o\|^2_{\bar{\boldsymbol{H}}^{(0)}\bar{\boldsymbol{H}}^{(1)}\cdots\bar{\boldsymbol{H}}^{(n)}\bar{\sigma}} + \bar{\boldsymbol{b}}^{(0)*}\bar{\boldsymbol{H}}^{(1)}\bar{\boldsymbol{H}}^{(2)}\cdots\bar{\boldsymbol{H}}^{(n)}\bar{\sigma}. \end{aligned} \tag{59}$$

By summing all the above equations, we have

$$E\left\|\bar{\boldsymbol{w}}^{(n)}\right\|^2_{\bar{\sigma}} \approx ||\bar{w}_o||^2_{\bar{\boldsymbol{H}}^{(0)}\bar{\boldsymbol{H}}^{(1)}\cdots\bar{\boldsymbol{H}}^{(n)}\bar{\sigma}} + \left[\bar{\boldsymbol{b}}^{(n)} + \sum_{m=1}^{n}\bar{\boldsymbol{b}}^{(m-1)*}\bar{\boldsymbol{H}}^{(m)}\bar{\boldsymbol{H}}^{(m+1)}\cdots\bar{\boldsymbol{H}}^{(n)}\right]\bar{\sigma}. \tag{60}$$

Following the equation above, we finally obtain the following recursion for the weighted mean-square of weight error vector:

$$\begin{aligned} E\left\|\bar{\boldsymbol{w}}^{(n)}\right\|^2_{\bar{\sigma}} &\approx E\left\|\bar{\boldsymbol{w}}^{(n-1)}\right\|^2_{\bar{\sigma}} - ||\bar{w}_o||^2_{\left[\mathcal{H}^{(n-1)} - \mathcal{H}^{(n)}\right]\bar{\sigma}} \\ \mathcal{H}^{(n)} &= \begin{cases} \mathcal{H}^{(n-1)}\mathcal{H}^{(n)}, & \text{for } n \geq 0 \\ I_{N^2L^2}, & \text{for } n = -1 \end{cases} \\ \mathcal{G}^{(n)} &= \begin{cases} \mathcal{G}^{(n-1)}\bar{\boldsymbol{H}}^{(n)} + \bar{\boldsymbol{b}}^{(n)}, & \text{for } n \geq 0 \\ 0_{N^2L^2}, & \text{for } n = -1. \end{cases} \end{aligned} \tag{61}$$

With Assumptions 1, 2 and 3, and based on Eq. (61), we have the following theorem:

Theorem 2. *(Learning curves of MSD and EMSE for the CR-dLMS algorithm) Let* $q = vec\{I_{NL}\}/N$ *and* $\lambda = vec\{\Lambda\}/N$*, where the diagonal matrix* Λ *comes from the decomposition* $R_x = T\Lambda T^*$*. The global MSD,* $\eta^{(n)}$*, and the global EMSE,* $\zeta^{(n)}$*, then evolve according to the following recursion:*

$$\eta^{(n)} = \eta^{(n-1)} - ||\bar{w}_o||^2_{\left[\mathcal{H}^{(n-1)} - \mathcal{H}^{(n)}\right]q} + \left[\mathcal{G}^{(n)} - \mathcal{G}^{(n-1)}\right]q, \tag{62}$$

$$\zeta^{(n)} = \zeta^{(n-1)} - ||\bar{w}_o||^2_{\left[\mathcal{H}^{(n-1)} - \mathcal{H}^{(n)}\right]\lambda} + \left[\mathcal{G}^{(n)} - \mathcal{G}^{(n-1)}\right]\lambda, \tag{63}$$

where initial global MSD and EMSE are given by $\eta^{(-1)} = \|w_o\|^2$ *and* $\zeta^{(-1)} = \|\bar{w}^o\|^2_\Lambda/N$*, respectively. The matrices* $\mathcal{H}^{(n)}$ *and* $\mathcal{G}^{(n)}$ *are recursively updated using Eq. (61).*

3.3. Learning behavior of our proposed CR-dLMS algorithm

In both the mean and mean-square behaviors of weight error vector, the matrix $\boldsymbol{S}^{(n)}$, which is relevant to the probability $\boldsymbol{p}_{lk}^{(n)}$ of data fusion from neighboring nodes, should be updated at each iteration. To facilitate learning behavior analysis, we replace the random variables in Eq. (24) with their expectations. Then, approximating the operation $E[res(\bullet)]$ with the operation $res[E(\bullet)]$ yields the following:

$$E\left[\boldsymbol{p}_k^{(n)}\right] = res\left\{E\left[\boldsymbol{p}_k^{(n-1)}\right] - \varepsilon_k E\left[\nabla_{\boldsymbol{p}_k} f\left(\boldsymbol{p}_k^{(n-1)}\right)\right]\right\}, \tag{64}$$

where

$$E\left[\nabla_{\boldsymbol{p}_k} f\left(\boldsymbol{p}_k^{(n-1)}\right)\right] = c_{lk}^2 E\left[\left\|\hat{\boldsymbol{w}}_{ok}^{(n-1)} - \hat{\psi}_{lk}^{(n-1)}\right\|^2\right] + \left\{-2c_{lk}\left(1 - \sum_{l\in\mathcal{N}_k\setminus k} E\left[\boldsymbol{p}_{lk}^{(n-1)}\right]c_{lk}\right) + \left(1 - 2E\left[\boldsymbol{p}_{lk}^{(n-1)}\right]\right)c_{lk}^2\right\} * E\left[\left\|\hat{\boldsymbol{w}}_{ok}^{(n-1)} - \boldsymbol{\psi}_k^{(n-1)}\right\|^2\right]. \tag{65}$$

To calculate Eq. (65), the terms

$$E\left[\left\|\hat{\boldsymbol{w}}_{ok}^{(n-1)} - \hat{\psi}_{lk}^{(n-1)}\right\|^2\right] \text{ and } E\left[\left\|\hat{\boldsymbol{w}}_{ok}^{(n-1)} - \boldsymbol{\psi}_k^{(n-1)}\right\|^2\right] \tag{66}$$

should be evaluated. Before evaluating these two terms, we make the following assumption, which has also been adopted in the literature of adaptive filtering [2,40,43]:

Assumption 4. At any iteration n, the error $\boldsymbol{\varepsilon}_k^{(n)} = \boldsymbol{x}_{k,n}\tilde{\boldsymbol{w}}_k^{(n-1)}$ is spatially independent and independent of the regressor vector $\boldsymbol{x}_{k,\,n}$.

Using Eq. (18) and using the approximation $\tilde{\boldsymbol{w}}_k^{(n-2)} \approx \tilde{\boldsymbol{\psi}}_k^{(n-1)}$, we have

$$\begin{aligned} E\left[||\hat{\boldsymbol{w}}_{ok}^{(n-1)} - \boldsymbol{\psi}_k^{(n-1)}||^2\right] &= E\left\{\left\|\mu_k \boldsymbol{x}_{k,n-1}^*\left[\boldsymbol{d}_k(n-1) - \boldsymbol{x}_{k,n-1}\boldsymbol{\psi}_k^{(n-1)}\right]\right\|^2\right\} \\ &= E\left\{\left\|\mu_k \boldsymbol{x}_{k,n-1}^*\left[-\boldsymbol{x}_{k,n-1}\tilde{\boldsymbol{\psi}}_k^{(n-1)} + \boldsymbol{z}_k(n-1)\right]\right\|^2\right\} \\ &\approx E\left\{\left\|\mu_k \boldsymbol{x}_{k,n-1}^*\left[-\boldsymbol{x}_{k,n-1}\tilde{\boldsymbol{w}}_k^{(n-2)} + \boldsymbol{z}_k(n-1)\right]\right\|^2\right\} \\ &= \mu_k^2 Tr\left(R_{x,k}\right)\left[\zeta_k^{(n-1)} + \sigma_{z,k}^2\right]. \end{aligned} \tag{67}$$

For the term $E[\|\hat{\boldsymbol{w}}_{ok}^{(n-1)} - \hat{\psi}_{lk}^{(n-1)}\|^2]$, we have

$$\begin{aligned} E\left[||\hat{\boldsymbol{w}}_{ok}^{(n-1)} - \hat{\psi}_{lk}^{(n-1)}||^2\right] &= E\left[\left\|\boldsymbol{\psi}_k^{(n-1)} - \hat{\psi}_{lk}^{(n-1)} + \mu_k \boldsymbol{x}_{k,n-1}^*\left[\boldsymbol{d}_k(n-1) - \boldsymbol{x}_{k,n-1}\boldsymbol{\psi}_k^{(n-1)}\right]\right\|^2\right] \\ &= E\left[\left\|\boldsymbol{\psi}_k^{(n-1)} - \hat{\psi}_{lk}^{(n-1)}\right\|^2\right] \\ &\quad -2E\left\{\left[\boldsymbol{\psi}_k^{(n-1)} - \hat{\psi}_{lk}^{(n-1)}\right]^*\left[\mu_k \boldsymbol{x}_{k,n-1}^*\left(\boldsymbol{d}_k(n-1) - \boldsymbol{x}_{k,n-1}\boldsymbol{\psi}_k^{(n-1)}\right)\right]\right\} \\ &\quad +\mu_k^2 E\left\{\left\|\boldsymbol{x}_k^*(n-1)\left[\boldsymbol{d}_k(n-1) - \boldsymbol{x}_{k,n-1}\boldsymbol{\psi}_k^{(n-1)}\right]\right\|^2\right\}. \end{aligned} \tag{68}$$

In the expression on RHS of the second equal sign in Eq. (68), the first term is approximated by the following transformations:

$$E\left[\left\|\boldsymbol{\psi}_k^{(n-1)}-\widehat{\psi}_{lk}^{(n-1)}\right\|^2\right]\approx\left\|\boldsymbol{\psi}_k^{(n-1)}-\widehat{\psi}_{lk}^{(n-1)}\right\|^2\approx\left\|\boldsymbol{\psi}_k^{(n-1)}-\boldsymbol{\psi}_l^{(n-1)}\right\|^2\approx\left\|\boldsymbol{w}_k^{(n-1)}-\boldsymbol{w}_l^{(n-1)}\right\|^2, \tag{69}$$

where the first "≈ " sign indicates that the expectation value is approximated by its instantaneous value. In the second "≈ " sign, $\widehat{\psi}_{lk}^{(n-1)}$ is approximated by $\boldsymbol{\psi}_l^{(n-1)}$. Finally, in the third "≈ " sign, $\boldsymbol{\psi}_k^{(n-1)}$ and $\boldsymbol{\psi}_l^{(n-1)}$ are approximated by $\boldsymbol{w}_k^{(n-1)}$ and $\boldsymbol{w}_l^{(n-1)}$ respectively. This approximation is reasonable under the sufficiently small step-size μ_k.

The second term of Eq. (68) is approximately expressed by

$$\begin{aligned}
&E\left\{\left[\boldsymbol{\psi}_k^{(n-1)}-\widehat{\psi}_{lk}^{(n-1)}\right]^*\ast\left[\mu_k\boldsymbol{x}_{k,n-1}^*\left(\boldsymbol{d}_k(n-1)-\boldsymbol{x}_{k,n-1}\boldsymbol{\psi}_k^{(n-1)}\right)\right]\right\}\\
&\approx\left[\boldsymbol{w}_k^{(n-1)}-\boldsymbol{w}_l^{(n-1)}\right]^*\ast E\left[\mu_k\boldsymbol{x}_{k,n-1}^*\left(\boldsymbol{d}_k(n-1)-\boldsymbol{x}_{k,n-1}\boldsymbol{\psi}_k^{(n-1)}\right)\right]\\
&\approx\mu_k\left[\boldsymbol{w}_k^{(n-1)}-\boldsymbol{w}_l^{(n-1)}\right]^*\ast E\left[\boldsymbol{x}_{k,n-1}^*\left(-\boldsymbol{x}_{k,n-1}\tilde{\boldsymbol{w}}_k^{(n-1)}+\boldsymbol{z}_k(n-1)\right)\right]\\
&=-\mu_k\left[\boldsymbol{w}_k^{(n-1)}-\boldsymbol{w}_l^{(n-1)}\right]^*\left[R_{x,k}\tilde{\boldsymbol{w}}_k^{(n-1)}\right].
\end{aligned} \tag{70}$$

Then, using Eq. (67), we can finally approximate Eq. (68) in the following:

$$\begin{aligned}
&E\left[||\hat{\boldsymbol{w}}_{ok}^{(n-1)}-\widehat{\psi}_{lk}^{(n-1)}||^2\right]\\
&\approx\left\|\boldsymbol{w}_k^{(n-1)}-\boldsymbol{w}_l^{(n-1)}\right\|^2+\mu_k^2Tr\left(R_{x,k}\right)\left[\zeta_k^{(n-1)}+\sigma_{z,k}^2\right]+2\mu_k\left[\boldsymbol{w}_k^{(n-1)}-\boldsymbol{w}_l^{(n-1)}\right]^*\left[R_{x,k}\tilde{\boldsymbol{w}}_k^{(n-1)}\right].
\end{aligned} \tag{71}$$

Finally, the process of tracking learning behavior of the CR-dLMS algorithm can be summarized by the following theorem:

Theorem 3. *Initialize* $\boldsymbol{w}_k^{(-1)}=0$ *and* $\boldsymbol{\psi}_k^{(-1)}=0$ *for any node k, and set* $E[\boldsymbol{p}_{lk}^{(-1)}]=\boldsymbol{p}_{lk}^{(-1)}=1$ *as the initial probability of node k fusing data from node l. Set* $\widehat{\psi}_{lk}(-1)=0$ *for any node k and its neighboring node l. The initial* $\zeta_k^{(-1)}$ *is set the same as that in Theorem 2. Then, by repeating the following steps, we can obtain the learning behavior of the CR-dLMS algorithm:*

1. *For each node k and its neighboring node l, calculate* $E[||\hat{\boldsymbol{w}}_{ok}^{(n-1)}-\boldsymbol{\psi}_k^{(n-1)}||^2]$ *according to Eq. (67), and* $E[||\hat{\boldsymbol{w}}_{ok}^{(n-1)}-\widehat{\psi}_{lk}^{(n-1)}||^2]$ *according to Eq. (71).*
2. *For each node k, calculate* $E\left[\nabla_{\boldsymbol{p}_k}f\left(\boldsymbol{p}_k^{(n-1)}\right)\right]$ *according to Eq. (65) and obtain the updated* $E\left[\boldsymbol{p}_k^{(n)}\right]$ *from* $E\left[\boldsymbol{p}_k^{(n-1)}\right]$ *according to Eq. (64).*
3. *Let* $E\left[\boldsymbol{p}_k^{(n)}\right]=\boldsymbol{p}_k^{(n)}$. *Calculate* $\bar{\boldsymbol{H}}^{(n)}$ *according to Eq. (56), and* $\bar{\boldsymbol{b}}^{(n)}$ *according to Eq. (A.2), (A.3), (A.4), (A.5) and (58).*
4. *Update EMSE and obtain* $\zeta_k^{(n)}$ *for each node k according to Eq. (63). Update* $\boldsymbol{w}_k^{(n)}$ *for each node according to the Eq. (38).*

4. Simulation results

In this section, we shall evaluate the performances of the CR-dLMS algorithm in multi-task environment by comparing it against several other algorithms, including the traditional ATC dLMS, the non-cooperative LMS and the dLMS with clustering strategy proposed in [8]. The traditional ATC dLMS algorithm is described by Eqs. (4) and (5). We consider both stationary environment and non-stationary environment here. In the stationary environment, the true weight vector on each node does not vary with time iterations; in the non-stationary environment, the true weight vector may vary with time iterations, thus reformulating clustering structure.

In our experiments on the traditional ATC dLMS and CR-dLMS algorithms, the combination coefficients were set according to the metropolis rule [46,49], i.e.

$$\begin{cases}
c_{lk}=1/\max\left(n_k,n_l\right), & l\in\mathcal{N}_k,\ l\neq k,\\
c_{lk}=0, & l\notin\mathcal{N}_k,\\
c_{lk}=1-\sum_{l\in\mathcal{N}_k\backslash k}c_{lk}, & l=k.
\end{cases} \tag{72}$$

Furthermore, each result in numerical simulations is the average over 300 independent simulation runs.

4.1. Stationary environment

First, we consider multi-task diffusion algorithms in stationary environment with four clusters shown in Fig. 2. Nodes filled in the same color belong to the same cluster, and different clusters are distinguished by different colors. The true

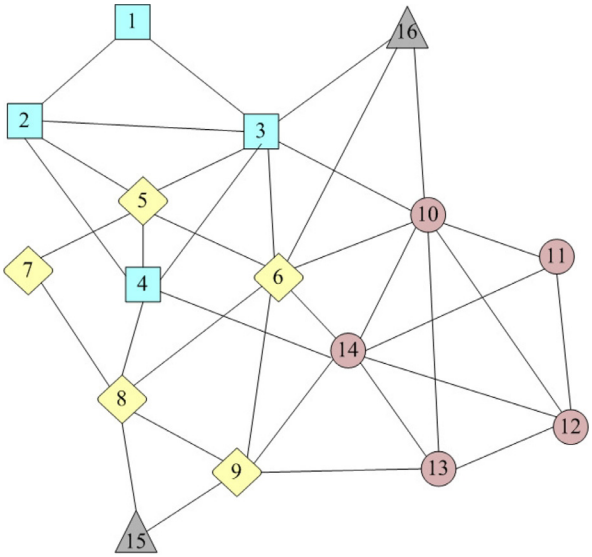

Fig. 2. Network topology with $N = 16$ nodes and clustering structure in the stationary environment.

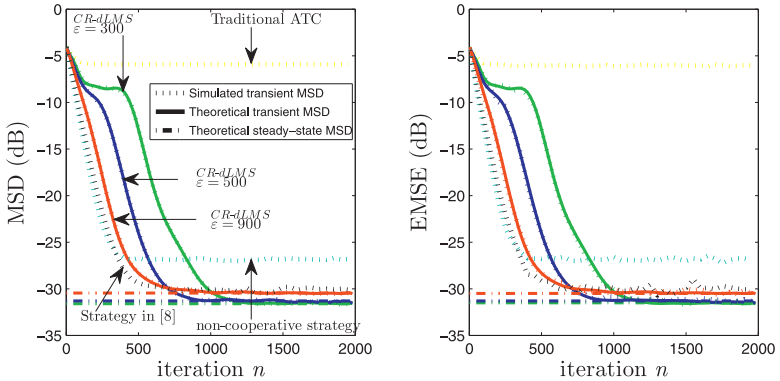

Fig. 3. Network MSD (a) and EMSE (b) behaviors in stationary environment. Results for both numerical simulations and theoretical analysis are represented by *dashed* and *solid* lines. In the CR-dLMS algorithm, the step-sizes ε_k for adapting data fusion probability are identical for all nodes.

weight vectors of all nodes are listed in the following:

$$w_{ok} = \begin{cases} [0.5 \quad -0.4]^T & k = 1, \ldots, 4 \quad \text{cluster 1}, \\ [0.6 \quad -0.2]^T & k = 5, \ldots, 9 \quad \text{cluster 2}, \\ [0.3 \quad -0.3]^T & k = 10, \ldots, 14 \quad \text{cluster 3}, \\ [-0.8 \quad 0.5]^T & k = 15, \ 16 \quad \text{cluster 4}. \end{cases} \tag{73}$$

Besides, the 1×2 regression vector $\boldsymbol{x}_{k,n}$ is governed by a Gaussian distribution with zero-mean and diagonal covariance matrix $R_{x,k} = \sigma_{x,k}^2 I_2$. For simplicity, $\sigma_{x,k}^2 = 1$ is identical for all nodes in the network. The variance of additive noise is also identical for all nodes, i.e. $\sigma_{z,k}^2 = 0.2, \ k = 1, \ldots, N$. Finally, the step-sizes in the adaptation process are set to $\mu_k = 0.01$ for all nodes in the network.

Fig. 3 illustrates the network transient MSD and EMSE for all algorithms under study in the stationary environment. We can observe that for the CR-dLMS algorithm, the theoretical and simulation results are accurately superimposed, and the theoretical steady-steady results are also observed to be consistent with steady-state values from numerical simulations. Due to the large bias among nodes from different clusters, the traditional ATC strategy achieves the largest MSD (or EMSE). Although the non-cooperative strategy converges without bias toward the true parameter, the strategy still fails to achieve satisfying MSD (or EMSE) level in multi-task environment. Compared to the clustering strategy in [8], the CR-dLMS can provide lower MSD (or EMSE) at the sacrifice of lower convergence rate of MSD (or EMSE). Besides, we have also found that the increase in ε_k leads to higher convergence rate of MSD (or EMSE). As shown in Fig. 3, convergence rate of MSD (or EMSE) for $\varepsilon_k = 900$ differs only slightly from that of the strategy in [8]. In a word, Fig. 3 reveals that the CR-dLMS algorithm

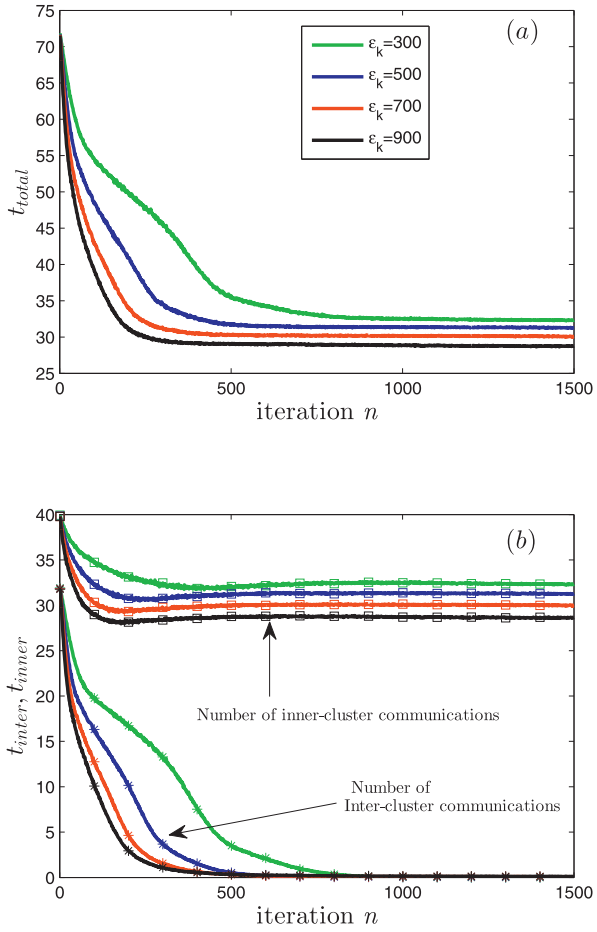

Fig. 4. The number of communications at each iteration for $\varepsilon_k = 300$, 500, 700 and 900. (a). The number of all communications at each iteration; (b). The number of inner-cluster communications and inter-cluster communications at each iteration.

can provide approximately the same performance as the strategy in [8] in regards to both the steady-value and convergence rate of MSD (or EMSE).

We next focus on the number of communications for the strategy in [8] and the CR-dLMS algorithm. As stated above, in the clustering strategy in [8], for any node k, the dynamical combination coefficients are calculated by collecting data from all its neighboring nodes at each iteration. However, in the CR-dLMS algorithm, each node does not need to cooperate with all its neighboring nodes at each iteration to judge which of them belong to the same cluster as itself, because cooperation in the CR-dLMS algorithm is controlled probabilistically. In the given network (shown in Fig. 2), there are 72 links in total, of which 40 are inner-cluster links, i.e. both ends of the link are in the same cluster. The remaining 32 links are inter-cluster links, i.e. the two ends of the link are from different clusters. Accordingly, at each iteration, the number of all communications is exactly 72 when using the clustering strategy in [8]. However, in the CR-dLMS algorithm, the number of total communications at each iteration is reduced to below 35 finally, as shown in Fig. 4(a). In addition, from Fig. 4(b), we find that the number of inter-cluster communications finally shrinks to zero, which indicates that under the CR-dLMS algorithm, each node can adaptively choose to cooperate with its neighboring nodes only through inner-cluster links. Inter-cluster communications are gradually filtered out with the adaptation process.

4.2. Non-stationary environment (1)

Now, we evaluate the performances of the CR-dLMS algorithm in non-stationary environment. The network topology is the same as the stationary case. In this and the next examples, we consider two scenarios under the non-stationary environment. In this section, the dynamical true parameters on nodes from the two clusters are shown in the following:

$$w_{ok}(n) = \begin{cases} [0.3 \ \ -0.2]^T + \frac{n-1000}{500}[0.2 \ \ -0.1]^T & k = 1, \ldots, 9 \quad \text{cluster 1,} \\ [0.3 \ \ -0.2]^T + \frac{n-1000}{500}[-0.1 \ \ 0.1]^T & k = 10, \ldots, 16 \quad \text{cluster 2.} \end{cases} \tag{74}$$

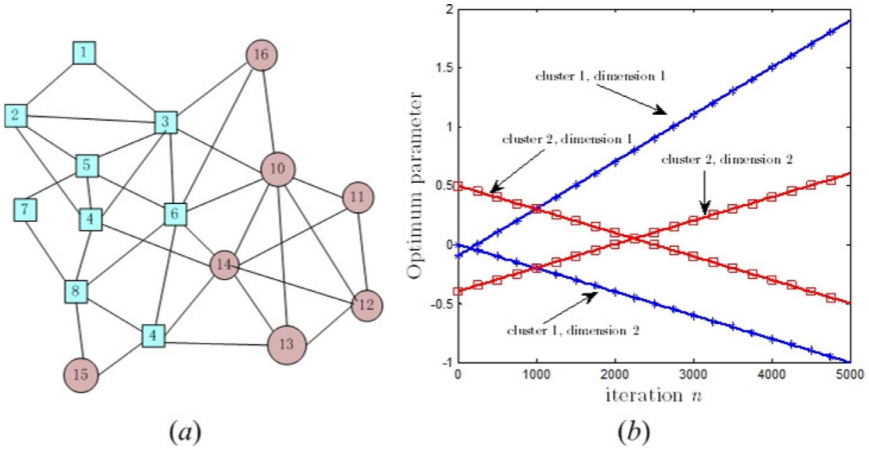

Fig. 5. (a). Network topology and clustering structure and (b). Evolution of true weight vectors for the non-stationary environment (1).

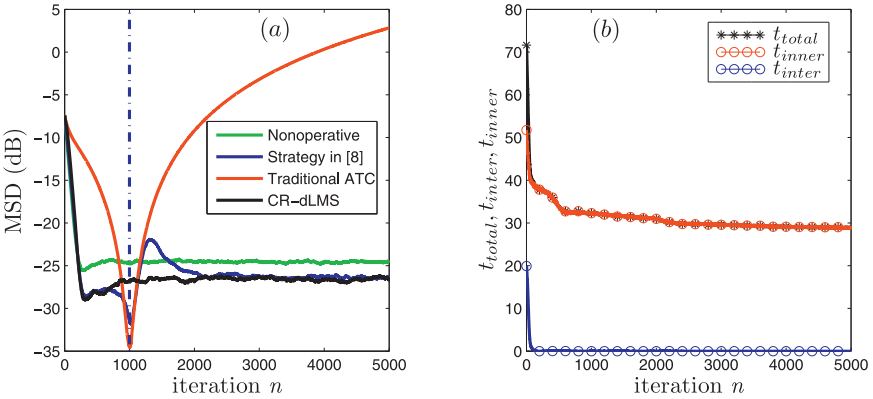

Fig. 6. (a). Network MSD behavior and (b). The number of total, inter-cluster and inner-cluster communications at each iteration in non-stationary environment (1).

The two clusters distinguished by different colors and the evolution of true parameters are illustrated in Fig. 5. As shown in Fig. 5(b), from iteration 1 to 1000, the true parameters of the two clusters gradually come nearer to each other. At iteration 1000, the two true parameters coincide with each other; from iteration 1000 on, the two true parameters grow farther away from each other with iterations. Properties of regression vector and noise in this experiment are kept the same as those used in the stationary case, and the parameter ε_k in the CR-dLMS algorithm is set to 1000.

The network MSD learning curves for the non-cooperative strategy, the strategy in [8], the traditional ATC dLMS strategy and the CR-dLMS algorithm are illustrated in Fig. 6(a). Two phases are observed from the MSD curve for the traditional ATC dLMS strategy. From iteration 1 to 1000, MSD level continually decrease as the true parameters of the two clusters grow nearer to each other. From iteration 1000 on, however, MSD level increases because parameter estimates on all nodes converge to the same solution in the traditional ATC dLMS strategy. Compared to the non-cooperative strategy, the CR-dLMS algorithm shows lower MSD level. Moreover, we have also found abrupt changes in the network MSD learning curve for the strategy in [8], but not in that of the CR-dLMS algorithm. This indicates that the proposed CR-dLMS algorithm can better adapt to the dynamical environment for parameter estimation in multi-task environment than the strategy in [8].

While the MSD level of the CR-dLMS algorithm does not deviate much from that of the strategy in [8], the number t_{total} of total communications at each iteration is much lower. As shown in Fig. 6(b), t_{total} decreases quickly with iterations and finally reaches stability at $t_{total} \approx 29$ in the CR-dLMS algorithm, covering only about 40% of that in the strategy in [8]. We have also observed that t_{inter}, i.e. the number of inter-cluster communications at each iteration, shrinks to zero very quickly. This result indicates that, in non-stationary environment, the CR-dLMS algorithm can achieve satisfying MSD level with much less communication cost.

4.3. Non-stationary environment (2)

We next consider a more complex scenario under the non-stationary environment, which has been explored in [8]. This scenario includes four stationary stages and three transient episodes. In each transient episode, the true weight vector is

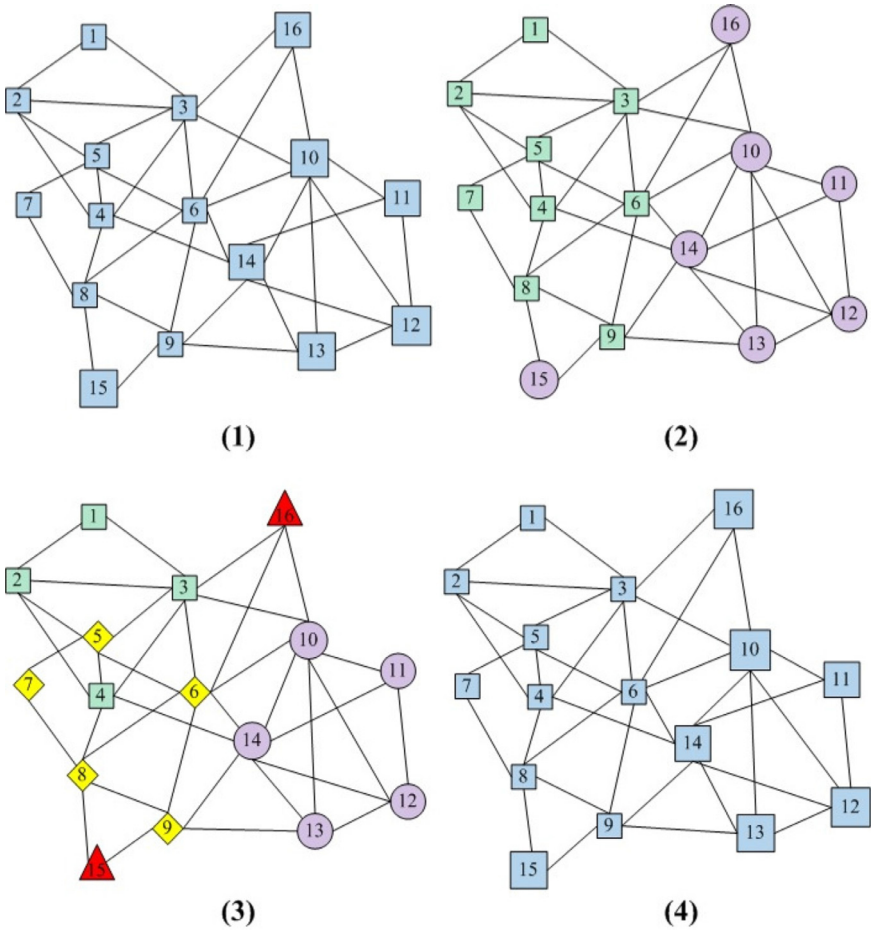

Fig. 7. Network structure and evolution of clustering structure for the non-stationary environment (2).

designed with linear interpolation between each steady-state stage over a period of 500 time samples. Time evolutions of clustering structure and true weight vectors are demonstrated in Figs. 7 and 8 respectively.

Fig. 9(a) shows the time evolutions of network MSD. The parameter ε_k in the CR-dLMS algorithm is set to 10,000 for nodes in the network in this test. In the stationary stages, the CR-dLMS algorithm provides higher MSD level than the algorithm in [8], but lower MSD level than the non-cooperative strategy. During transient episodes, however, the CR-dLMS algorithm outperforms the algorithm in [8] in regards to the MSD learning curve. The traditional ATC dLMS algorithm, outperforms all the other algorithms considered here if only one cluster exists in the network; if there is more than one cluster in the network, said performance rapidly deteriorates. This is because the tradition ATC dLMS algorithm enforces all parameter estimates on each node to converge to the same solution.

We next focus on the communication cost consumed by cooperation among nodes in the CR-dLMS algorithm, which is illustrated in Fig. 9(b). Note that in the case where only one cluster exists in the network, all communications are regarded as inner-cluster communications. Each node in this case only cooperates with part of its neighboring nodes under the CR-dLMS algorithm, which leads to a higher MSD level under the CR-dLMS algorithm than under the traditional ATC dLMS algorithm. When there are two clusters at time $t = 1000$, or four at time $t = 2500$, the number t_{inter} of inter-cluster communications increases as there are more inter-cluster communications. However, t_{inter} gradually decreases after the transition moment. This result again confirms that under the CR-dLMS algorithm, the inter-cluster communications can be adaptively filtered out for distributed parameter estimation in multi-task environment without sacrificing estimation performance.

4.4. Multiple target tracking over sensor networks

Finally, we consider the problem of collaborative tracking over sensor network (as likewise addressed in [8]). In this problem, four targets, numbered from $i = 1$ to 4, move in a two-dimensional plane according to the following position-transition equation:

$$\boldsymbol{x}_i(n+1) = T_i \boldsymbol{x}_i(n) + \boldsymbol{z}_i(n), \tag{75}$$

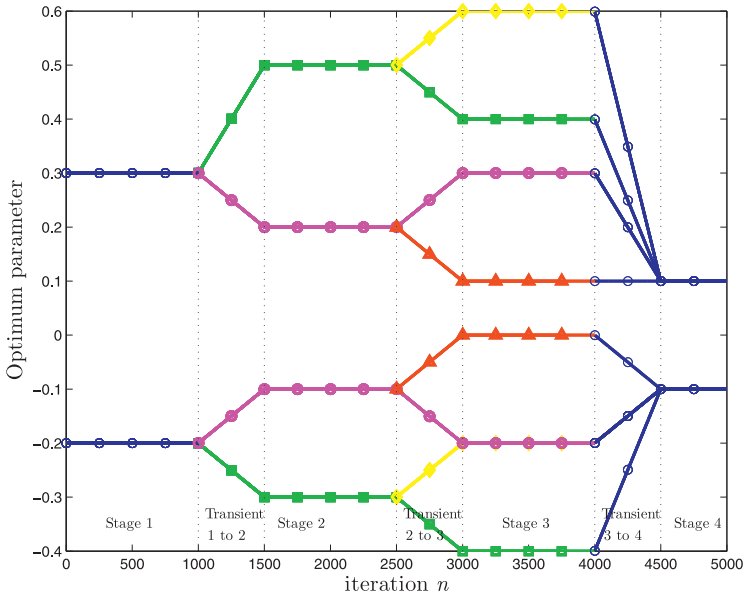

Fig. 8. Evolution of true weight vectors for the non-stationary environment (2).

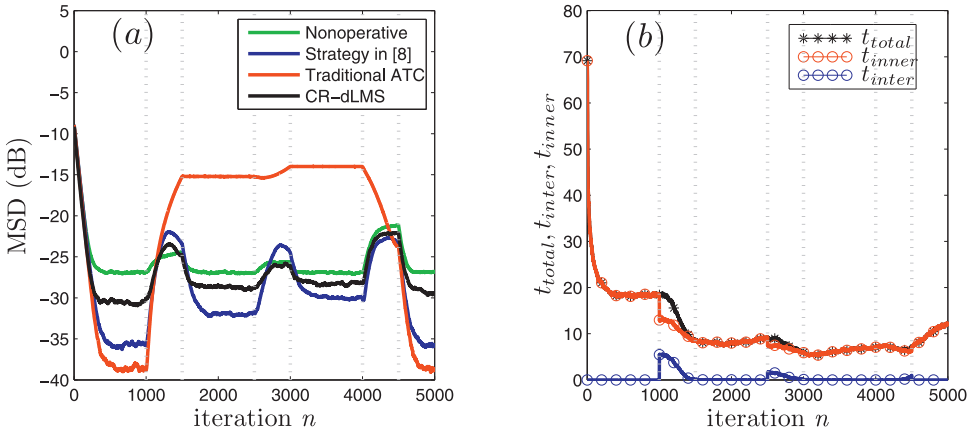

Fig. 9. (a). Network MSD behavior and (b). The number of total, inter-cluster and inner-cluster communications at each iteration in non-stationary environment (2).

in which $\boldsymbol{x}_i(n)$ denotes the position of target i at iteration n, and $\boldsymbol{z}_i$ is the modeling noise with i.i.d zero-mean Gaussian distribution with covariance $\sigma_z^2 I_2 = 0.05^2 I_2$. The target-dependent parameters T_i are the 2×2 position-transition matrices, which are set to

$$T_1 = \begin{pmatrix} 1.0019 & -0.0129 \\ 0.0187 & 1.0034 \end{pmatrix}, \; T_2 = \begin{pmatrix} 1.0149 & -0.0014 \\ 0.0033 & 1.0034 \end{pmatrix}, \\ T_3 = \begin{pmatrix} 1.0128 & -0.0041 \\ 0.0156 & 1.0086 \end{pmatrix}, \; T_4 = T_1. \tag{76}$$

Now, we consider a connected random network with $N = 100$ nodes deployed in a given area, wherein each node tracks only the noised position of one target. The random network was generated by connecting each pair of nodes using a link with an identical probability $p = 0.1$ [12]. Due to measurement error, the noised position is given by

$$\tilde{\boldsymbol{x}}_i(n) = \boldsymbol{x}_i(n) + \boldsymbol{u}_i(n), \; i = 1, 2, 3, 4, \tag{77}$$

in which $\boldsymbol{u}_i(n)$ is the measurement noise with i.i.d zero-mean Gaussian distribution with covariance $\sigma_u^2 I_2 = 0.05^2 I_2$. Here, we assume nodes 1 to 25 track target 1, nodes 26 to 50 track target 2, nodes 51 to 75 track target 3 and nodes 76 to 100 track target 4.

Regarding $\tilde{\boldsymbol{x}}_i(n)$ as the input data and $\tilde{\boldsymbol{x}}_i(n+1)$ as the output data, the position-transition matrices of all targets can be estimated under the distributed diffusion theory for parameter estimation over sensor networks. The non-cooperative

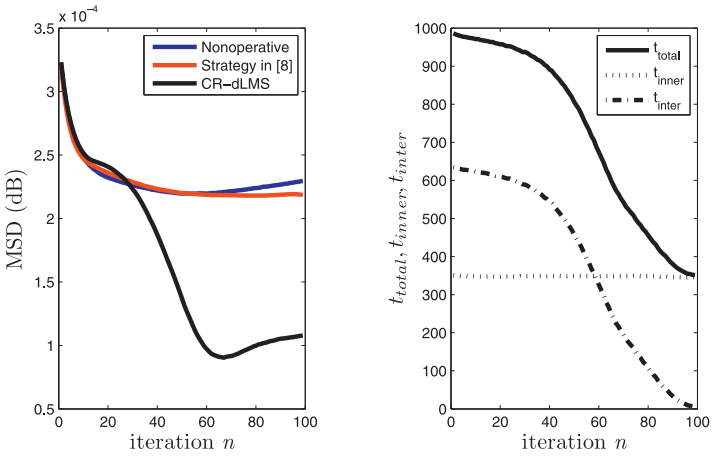

Fig. 10. (a). Network MSD behavior for the problem of multiple target tracking. (b). The number of total, inter-cluster and inner-cluster communications at each iteration. In the CR-dLMS algorithm, the step-size for adapting data fusion probability is set to $\varepsilon_k = 10^5$. The step-sizes for all three algorithms are set to $\mu_k = 10^5$.

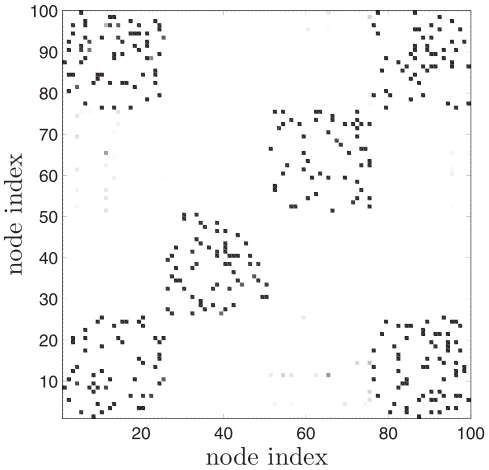

Fig. 11. Matrix of data fusion probability resulting from our proposed CR-dLMS algorithm at iteration $n = 100$.

strategy, the clustering strategy in [8], and our proposed CR-dLMS algorithm, were respectively tested in terms of their ability to estimate the position-transition matrices of all targets. MSD learning curves for all algorithms under study are plotted in Fig. 10(a). It can be clearly observed that, the non-cooperative strategy and the strategy in [8] deliver similar MSD level. Compared to these two algorithms, the CR-dLMS algorithm delivers much lower MSD level.

We next examine the number of communications in the CR-dLMS algorithm. There are 988 links in total in the random network as generated, including 352 inner-cluster links and 636 inter-cluster links. Fig. 10(b) illustrates the iteration evolution of t_{total}, t_{inner} and t_{inter} for the CR-dLMS algorithm, where t_{inner} keeps practically unchanged (around 350) as the iterations progress - in other words, almost all inner-cluster links are utilized for data exchange throughout the adaptive target tracking process. In the meanwhile, t_{inter} gradually decreases as the iterations elapse. At iteration $n = 100$, data exchange among nodes does not occur through inter-cluster links. Thus, as compared to the strategy in [8], the CR-dLMS algorithm can achieve satisfying performance in position-transition matrix estimation with much lower communication cost.

Fig. 11 illustrates the clustering ability of the CR-dLMS algorithm in more detail via the data fusion probability matrix at iteration $n = 100$. Darker elements in the matrix indicate larger data fusion probability in neighboring node pairs. We can observe that distinct clustering effect emerges, which validates the efficient clustering ability of the CR-dLMS algorithm.

The results from numerical simulations described above altogether suggest that in the context of multi-task parameter estimation, the main advantage of the proposed CR-dLMS algorithm is communication reduction; it succeeds in this regard while maintaining similar estimation performance compared to several other algorithms. In algorithms such as the traditional ATC dLMS algorithm, the algorithm proposed in [8], each node fuses parameter estimates from all its neighboring nodes at each iteration, including the neighboring nodes in the same cluster and the neighboring nodes from different clusters. Data fusion between nodes from different clusters does not in actuality enhance estimation performance, as the

performance of the CR-dLMS algorithm indicates. The CR-dLMS algorithm allows each node to choose to fuse data only from neighboring nodes in the same cluster at each iteration in an adaptive manner. This keeps communication cost low because there is low likelihood of data fusion between nodes from different clusters during adaptive parameter estimation.

5. Conclusion

In this work, we proposed an adaptive strategy, called the CR-dLMS algorithm, for controlling data fusion probability from neighboring nodes in multi-task orientated distributed diffusion LMS algorithm for parameter estimation. Compared to several other algorithms, the CR-dLMS algorithm was found to achieve satisfying estimation performance at greatly reduced communication cost. Both the mean weight behavior and mean-square deviation were derived. Finally, simulation results were presented to validate the efficiency of the proposed CR-dLMS algorithm.

Acknowledgments

This work was supported by National Natural Science Foundation of China (61201074, 61603339, 61325019), Zhejiang Provincial Natural Science Foundation of China (LY14F020018, LQ14F030003), HongKong, Macao and Taiwan Science & Technology Cooperation Program of China (2014DFH10110), and Science and Technique Program of Zhejiang Province (2015C31059).

Appendix A. Evaluation for the four terms in Eq. (49).

For evaluating the term $E[\boldsymbol{p}^{*}_{z\tilde{x},n}U\bar{\boldsymbol{K}}^{(n)}\bar{\Sigma}\bar{\boldsymbol{K}}^{(n)*}U\boldsymbol{p}_{z\tilde{x},n}]$, defining

$$\Lambda_z = diag\left\{\sigma^2_{z,1},\ldots,\sigma^2_{z,N}\right\}, \tag{A.1}$$

we have

$$\begin{aligned}
E\left[\boldsymbol{p}^{*}_{z\tilde{x},n}U\bar{\boldsymbol{K}}^{(n)}\bar{\Sigma}\bar{\boldsymbol{K}}^{(n)*}U\boldsymbol{p}_{z\tilde{x},n}\right] &= E\left[Tr\left(\bar{\boldsymbol{K}}^{(n)*}U\boldsymbol{p}_{z\tilde{x},n}\boldsymbol{p}^{*}_{z\tilde{x},n}U\bar{\boldsymbol{K}}^{(n)}\bar{\Sigma}\right)\right]\\
&= E\left[vec\left(\bar{\boldsymbol{K}}^{(n)*}U\boldsymbol{p}_{z\tilde{x},n}\boldsymbol{p}^{*}_{z\tilde{x},n}U\bar{\boldsymbol{K}}^{(n)}\right)^{*}\bar{\sigma}\right]\\
&= E\left[\left(\left(\bar{\boldsymbol{K}}^{(n)*}\otimes\bar{\boldsymbol{K}}^{(n)*}\right)vec\left(U\boldsymbol{p}_{z\tilde{x},n}\boldsymbol{p}^{*}_{z\tilde{x},n}U\right)\right)^{*}\right]\bar{\sigma}\\
&\approx\left\{\left[E\left(\bar{\boldsymbol{K}}^{(n)*}\right)\otimes E\left(\bar{\boldsymbol{K}}^{(n)*}\right)\right]vec\left[U^2\Lambda(\Lambda_z\otimes I_L)\right]\right\}^{*}\bar{\sigma}\\
&= \left\{\left[\bar{\boldsymbol{S}}^{(n)*}\otimes\bar{\boldsymbol{S}}^{(n)*}\right]vec\left[U^2\Lambda(\Lambda_z\otimes I_L)\right]\right\}^{*}\bar{\sigma}\\
&= \bar{\boldsymbol{b}}^{(n)*}_{1}\bar{\sigma}.
\end{aligned} \tag{A.2}$$

In the approximation $\approx$, we assume $E[(\bar{\boldsymbol{K}}^{(n)*}\otimes\bar{\boldsymbol{K}}^{(n)*})]\approx E[\bar{\boldsymbol{K}}^{(n)*}]\otimes E[\bar{\boldsymbol{K}}^{(n)*}]$. This approximation has been made in [43] and does not alter the conclusion too much.

For the term $E[\bar{w}^{*}_{o}(\bar{\boldsymbol{K}}^{(n)}-I_{NL})\bar{\Sigma}(\bar{\boldsymbol{K}}^{(n)*}-I_{NL})\bar{w}_o]$, we have

$$\begin{aligned}
&E\left[\bar{w}^{*}_{o}\left(\bar{\boldsymbol{K}}^{(n)}-I_{NL}\right)\bar{\Sigma}\left(\bar{\boldsymbol{K}}^{(n)*}-I_{NL}\right)\bar{w}_o\right]\\
&= E\left[Tr\left(\left(\bar{\boldsymbol{K}}^{(n)*}-I_{NL}\right)\bar{w}_o\bar{w}^{*}_{o}\left(\bar{\boldsymbol{K}}^{(n)}-I_{NL}\right)\bar{\Sigma}\right)\right]\\
&= E\left[vec\left(\left(\bar{\boldsymbol{K}}^{(n)*}-I_{NL}\right)\bar{w}_o\bar{w}^{*}_{o}\left(\bar{\boldsymbol{K}}^{(n)}-I_{NL}\right)\right)^{*}\bar{\sigma}\right]\\
&\approx\left\{\left[\left(\bar{\boldsymbol{S}}^{(n)*}-I_{NL}\right)\otimes\left(\bar{\boldsymbol{S}}^{(n)*}-I_{NL}\right)\right]vec(\bar{w}_o\bar{w}^{*}_{o})\right\}^{*}\bar{\sigma}\\
&= \bar{\boldsymbol{b}}^{(n)*}_{2}\bar{\sigma}.
\end{aligned} \tag{A.3}$$

For the term
$E[\bar{\boldsymbol{w}}^{(n-1)*}(I_{NL}-U\bar{\boldsymbol{x}}^{*}_{n}\bar{\boldsymbol{x}}_n)\bar{\boldsymbol{K}}^{(n)}\bar{\Sigma}[\bar{\boldsymbol{K}}^{(n)*}-I_{NL}]\bar{w}_o]$, we have

$$\begin{aligned}
&E\left[\bar{\boldsymbol{w}}^{(n-1)*}\left(I_{NL}-U\bar{\boldsymbol{x}}^{*}_{n}\bar{\boldsymbol{x}}_n\right)\bar{\boldsymbol{K}}^{(n)}\bar{\Sigma}\left[\bar{\boldsymbol{K}}^{(n)*}-I_{NL}\right]\bar{w}_o\right]\\
&\approx Tr\left[\left(\bar{\boldsymbol{S}}^{(n)*}-I_{NL}\right)\bar{w}_oE\left(\bar{\boldsymbol{w}}^{(n-1)*}\right)(I_{NL}-U\Lambda)\bar{\boldsymbol{S}}^{(n)}\bar{\Sigma}\right]\\
&= vec\left[\bar{\boldsymbol{S}}^{(n)*}(I_{NL}-U\Lambda)E\left(\bar{\boldsymbol{w}}^{(n-1)}\right)\bar{w}^{*}_{o}\left(\bar{\boldsymbol{S}}^{(n)}-I_{NL}\right)\right]^{*}\bar{\sigma}
\end{aligned}$$

$$= \left\{ \left[\left(\bar{\boldsymbol{S}}^{(n)*} - I_{NL} \right) \otimes \bar{\boldsymbol{S}}^{(n)*} \right] vec\left[(I_{NL} - U\Lambda) E\left(\bar{\boldsymbol{w}}^{(n-1)} \right) \bar{w}_o^* \right] \right\}^* \bar{\sigma}$$

$$= \bar{\boldsymbol{b}}_3^{(n)*} \bar{\sigma}. \tag{A.4}$$

For the term $E[\bar{w}_o^* (\bar{\boldsymbol{K}}^{(n)} - I_{NL}) \bar{\Sigma} \bar{\boldsymbol{K}}^{(n)*} (I_{NL} - U\bar{\boldsymbol{x}}_n^* \bar{\boldsymbol{x}}_n) \bar{\boldsymbol{w}}^{(n-1)}]$, we have

$$E\left[\bar{w}_o^* \left(\bar{\boldsymbol{K}}^{(n)} - I_{NL} \right) \bar{\Sigma} \bar{\boldsymbol{K}}^{(n)*} \left(I_{NL} - U\bar{\boldsymbol{x}}_n^* \bar{\boldsymbol{x}}_n \right) \bar{\boldsymbol{w}}^{(n-1)} \right]$$

$$\approx \bar{w}_o^* \left(\bar{\boldsymbol{S}}^{(n)} - I_{NL} \right) \bar{\Sigma} \bar{\boldsymbol{S}}^{(n)*} (I_{NL} - U\Lambda) E\left[\bar{\boldsymbol{w}}^{(n-1)} \right]$$

$$= Tr\left\{ \bar{\boldsymbol{S}}^{(n)*} (I_{NL} - U\Lambda) E\left[\bar{\boldsymbol{w}}^{(n-1)} \right] \bar{w}_o^* \left(\bar{\boldsymbol{S}}^{(n)} - I_{NL} \right) \bar{\Sigma} \right\}$$

$$= vec\left\{ \left(\bar{\boldsymbol{S}}^{(n)*} - I_{NL} \right) \bar{w}_o \, E\left[\bar{\boldsymbol{w}}^{(n-1)*} \right] (I_{NL} - U\Lambda) \bar{\boldsymbol{S}}^{(n)*} \right\}^* \bar{\sigma}$$

$$= \left\{ \left[\bar{\boldsymbol{S}}^{(n)} \otimes \left(\bar{\boldsymbol{S}}^{(n)*} - I_{NL} \right) \right] vec\left[\bar{w}_o E\left(\bar{\boldsymbol{w}}^{(n-1)*} \right) (I_{NL} - U\Lambda) \right] \right\}^* \bar{\sigma}$$

$$= \bar{\boldsymbol{b}}_4^{(n)*} \bar{\sigma}. \tag{A.5}$$

References

[1] R. Abdolee, B. Champagne, A.H. Sayed, Estimation of space-time varying parameters using a diffusion LMS algorithm, IEEE Trans. Signal Process. 62 (2) (2014) 403–418.
[2] T.Y. Al-Naffouri, A.H. Sayed, Transient analysis of adaptive filters with error nonlinearities, IEEE Trans. Signal Process. 51 (3) (2003) 653–663.
[3] N. Bogdanovic, J. Plata-Chaves, K. Berberidis, Distributed incremental-based LMS for node-specific adaptive parameter estimation, IEEE Trans. Signal Process. 62 (20) (2014) 5382–5397.
[4] X. Cao, J. Chen, Y. Xiao, Y. Sun, Building-environment control with wireless sensor and actuator networks: centralized versus distributed, IEEE Trans. Ind. Electron. 57 (11) (2010) 3596–3605.
[5] F.S. Cattivelli, C.G. Lopes, A.H. Sayed, Diffusion recursive least-squares for distributed estimation over adaptive networks, IEEE Trans. Signal Process. 56 (5) (2008) 1865–1877.
[6] F.S. Cattivelli, A.H. Sayed, Diffusion LMS strategies for distributed estimation, IEEE Trans. Signal Process. 58 (3) (2010) 1035–1048.
[7] J. Chen, C. Richard, A.O. Hero, A.H. Sayed, Diffusion LMS for multitask problems with overlapping hypothesis subspaces, in: 2014 IEEE International Workshop on Machine Learning for Signal Processing, September, 2014, pp. 21–24.
[8] J. Chen, C. Richard, A.H. Sayed, Diffusion LMS over multitask networks, IEEE Trans. Signal Process. 63 (11) (2014b) 2733–2748.
[9] J. Chen, A.H. Sayed, Diffusion adaptation strategies for distributed optimization and learning over networks, IEEE Trans. Signal Process. 60 (8) (2012) 4289–4305.
[10] S. Chouvardas, K. Slavakis, Y. Kopsinis, S. Theodoridis, A sparsity promoting adaptive algorithm for distributed learning, IEEE Trans. Signal Process. 60 (10) (2012) 5412–5425.
[11] S. Chouvardas, K. Slavakis, S. Theodoridis, Adaptive robust distributed learning in diffusion sensor networks, IEEE Trans. Signal Process. 59 (10) (2011) 4692–4707.
[12] P. Erdös, A. Rényi, On the strength of connectedness of a random graph, Acta Math. Acad. Sci. Hung. 12 (1) (1964) 261–267.
[13] A. Feuer, E. Weinstein, Convergence analysis of LMS filters with uncorrelated gaussian data, IEEE Trans. Acoust. Speech Signal Process. 33 (1) (1985) 222–230.
[14] X. Ge, Q.L. Han, Distributed event-triggered H_∞ filtering over sensor networks with communication delays, Inf. Sci. 291 (2015) 128–142.
[15] O.N. Gharehshiran, V. Krishnamurthy, G. Yin, Distributed energy-aware diffusion least mean squares: game-theoretic learning, IEEE J. Sel. Top. Signal Process. 7 (5) (2013) 1–16.
[16] V.C. Gogineni, M. Chakraborty, Diffusion adaptation over clustered multitask networks based on the affine projection algorithm, 2015. [Online]. Available: http://arxiv.org/abs/1507.08566.
[17] V.C. Gogineni, M. Chakraborty, Distributed multi-task APA over adaptive networks based on partial diffusion, 2015b. [Online]. Available: http://arxiv.org/abs/1509.09157.
[18] W. Huang, X. Yang, D. Liu, S. Chen, Diffusion LMS with component-wise variable step-size over sensor networks, IET Signal Processs. 10 (1) (2016) 37–45.
[19] L.L. Horowitz, K.D. Senne, Performance advantage of complex LMS for controlling narrow-band adaptive arrays, IEEE Trans. Acoust. Speech Signal Process. 29 (3) (1981) 722–736.
[20] M. Joseph, J.Q. Maguire, Cognitive radio: making software radios more personal, IEEE Pers. Commun. 6 (4) (1999) 13–18.
[21] S.M. Jung, J.H. Seo, P.G. Park, A variable step-size diffusion normalized least-mean-square algorithm with a combination method based on mean-square deviation, Circuits, Systems, and Signal Processing, Springer, February, 2015.
[22] A. Khalili, M.A. Tinati, A. Rastegarnia, Steady-state analysis of incremental LMS adaptive networks with noisy links, IEEE Trans. Signal Process. 59 (5) (2011) 2416–2422.
[23] A. Khalili, M.A. Tinati, A. Rastegarnia, J.A. Chambers, Steady-state analysis of diffusion LMS adaptive networks with noisy links, IEEE Trans. Signal Process. 60 (2) (2012) 974–979.
[24] S. Khawatmi, A.M. Zoubir, A.H. Sayed, Decentralized clustering over adaptive networks, in: IEEE 23rd European Signal Conference, EUSIPCO 2015, 2015, pp. 2746–2750.
[25] H.S. Lee, S.E. Kim, J.W. Lee, W.J. Song, A variable step-size diffusion LMS algorithm for distributed estimation, IEEE Trans. Signal Process. 63 (7) (2015) 1808–1820.
[26] C. Li, S. Huang, Y. Liu, Y. Liu, Distributed TLS over Multi-task Networks with Adaptive Inter-task Cooperation, in: IEEE Trans. Aerosp.Electron. Syst., in press.
[27] C. Li, P. Shen, Y. Liu Y, Z. Zhang, Diffusion information theoretic learning for distributed estimation over network, IEEE Trans. Signal Process. 61 (16) (2013) 4011–4024.
[28] Y. Liu, C. Li, W.K.S. Tang, Z. Zhang, Distributed estimation over complex networks, Inf. Sci. 197 (2012a) 91–104.
[29] Y. Liu, C. Li, Z. Zhang, Diffusion sparse least-mean squares over networks, IEEE Trans. Signal Process. 60 (8) (2012b) 4480–4485.
[30] Y. Liu, C. Yang, W.K.S. Tang, C. Li, Optimal topological design for distributed estimation over sensor networks, Inf. Sci. 254 (2014) 83–97.
[31] C.G. Lopes, A.H. Sayed, Incremental adaptive strategies over distributed networks, IEEE Trans. Signal Process. 55 (8) (2007) 4064–4077.

[32] C.G. Lopes, A.H. Sayed, Diffusion least-mean squares over adaptive networks: formulation and performance analysis, IEEE Trans. Signal Process. 56 (7) (2008) 3122–3136.
[33] P.D. Lorenzo, A.H. Sayed, Sparse distributed learning based on diffusion adaptation, IEEE Trans. Signal Process. 61 (6) (2012) 1419–1433.
[34] J. Mitola, G. Q.Maguire, Cognitive radio: making software radios more personal, IEEE Pers. Commun. 6 (1999) 13–18.
[35] J. Plata-Chaves, M.H. Bahari, M. Moonen, A. Bertrand, Unsupervised diffusion-based LMS for node-specific parameter estimation over wireless sensor networks, in: IEEE 41th International Conference on Acoustics, Speech and Signal Processing, 2016.
[36] J. Plata-Chaves, N. Bogdanovic, K. Berberidis, Distributed incremental-based RLS for node-specific parameter estimation over adaptive networks, in: IEEE 21st European Signal Conference, 2013. EUSIPCO, 2013.
[37] J. Plata-Chaves, N. Bogdanovic, K. Berberidis, Distributed diffusion-based LMS for node-specific adaptive parameter estimation, IEEE Trans. Signal Process. 13 (63) (2015) 3448–3460.
[38] C. Richard, A.H. Sayed, J. Chen, Multitask diffusion adaptation over networks, IEEE Trans. Signal Process. 62 (16) (2014) 4129–4144.
[39] M.O.B. Saeed, A. Zerguine, S.A. Zummo, A variable step size strategy for distributed estimation over adaptive networks, EURASIP J. Adv. Signal Process. 135 (1) (2013) 656–667.
[40] A.H. Sayed, Fundamentals of Adaptive Filtering, Wiley, New York, 2003.
[41] A.H. Sayed, Diffusion Adaptation over Networks, Academic Press Libraray in Signal Processing, New York, NY, USA: Elsevier, 2014, pp. 322–454.
[42] J. Shen, H. Tan, J. Wang, J. Wang, S. Lee, A novel routing protocol providing good transmission reliability in underwater sensor networks, J. Internet Technol. 16 (1) (2015) 171–178.
[43] N. Takahashi, I. Yamada, A.H. Sayed, Diffusion least-mean squares with adaptive combiners: formulation and performance analysis, IEEE Trans. Signal Process. 58 (9) (2010) 4795–4810.
[44] S.-Y. Tu, A.H. Sayed, Mobile adaptive networks, IEEE J. Sel. Top. Signal Process. 5 (4) (2011) 649–664.
[45] R.L. Valcarce, S.S. Pereira, A.P. Zamora, Distributed total least squares estimation over networks, in: ICASSP, 2014, pp. 7580–7584.
[46] H. Wang, X. Liao, T. Huang, G. Chen, Distributed parameter estimation in unreliable WSNs: quantized communication and asynchronous intermittent observation, Inf. Sci. 309 (2015) 11–25.
[47] Y. Wang, W.P. Tay, W. Wu, Multitask diffusion LMS with optimized inter-cluster cooperation, in: Proc. IEEE Workshop on Statistical Signal Processing, Palma de Mallorca, Spain, 2016.
[48] B. Widrow, J. Mccool, M.G. Larimore, C.R. Johnson, Stationary and nonstationary learning characteristics of the LMS adaptive filter, Proc. IEEE 64 (8) (1976) 1151–1162.
[49] L. Xiao, S. Boyd, Fast linear iterations for distributed averaging, Syst. Control Lett. 53 (1) (2004) 65–78.
[50] Y. Yang, Y. Liang, Q. Pan, Y. Qin, F. Yang, Distributed fusion estimation with square-root array implementation for markovian jump linear systems with random parameter matrices and cross-correlated noises, Inf. Sci. 370–371 (2016) 446–462.
[51] X. Zhao, A.H. Sayed, Clustering via diffusion adaptation over networks, in: Proceedings of the International Workshop on Cognitive Information Processing(CIP), Baiona, 2012, pp. 1–6. Spain
[52] X. Zhao, A.H. Sayed, Distributed clustering and learning over networks, IEEE Trans. Signal Process. 63 (13) (2015) 3285–3300.
[53] X. Zhao, S.Y. Tu, A.H. Sayed, Diffusion adaptation over networks under imperfect information exchange and non-stationary data, IEEE Trans. Signal Process. 60 (7) (2012) 3460–3475.